PENCARIAN RESEP MAKANAN BERDASARKAN CITRA MAKANAN MENGGUNAKAN EKSTRAKSI FITUR *SIMPLE MORPHOLOGICAL SHAPE DESCRIPTORS* DAN *COLOR MOMENT*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Tri Rahayuni
NIM: 155150201111010

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

PENGESAHAN

PENCARIAN RESEP MAKANAN BERDASARKAN CITRA MAKANAN MENGGUNAKAN EKSTRAKSI FITUR *SIMPLE MORPHOLOGICAL SHAPE DESCRIPTORS* DAN *COLOR MOMENT*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Tri Rahayuni
NIM: 155150201111010

Skripsi ini telah diuji dan dinyatakan lulus pada
2 Januari 2019
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Yuita Arum Sari, S.Kom., M.Kom	Sigit Adinugroho, S.Kom., M.Sc.
NIK: 2016098807152001	NIK: 201607 880701 1 000

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T., Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 2 Januari 2019

Tri Rahayuni

NIM: 155150201111010

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga laporan skripsi yang berjudul "Pencarian Resep Makanan Berdasarkan Citra Makanan Menggunakan Ekstraksi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*" ini dapat terselesaikan.

Penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Ibu Yuita Arum Sari, S.Kom., M.Kom dan Bapak Sigit Adinugroho, S.Kom., M.Sc. selaku Pembimbing skripsi yang telah dengan sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini.
2. Bapak Maksum dan Ibu Arba'inah selaku orang tua penulis dan seluruh keluarga besar atas segala nasihat, kasih sayang, perhatian dan kesabarannya di dalam membesarkan dan mendidik penulis, serta yang senantiasa tiada henti-hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
3. Teman-teman LPM DISPLAY FILKOM, penghuni grup #GABISAGASOMBONG dan penghuni grup Ka Depay yang telah memberikan banyak dukungan sekaligus pengalaman selama tiga tahun.
4. Teman-teman Teknik Informatika kelas A 2015, nelli, fat, choi, dilla, rosita, kris, tama yang telah mengisi hari-hari penulis selama menempuh pendidikan di FILKOM UB dan memberikan dukungan dan semangat kepada penulis.
5. Teman-teman kelas Induksi Riset kelas F yang selama ini telah saling bertukar ilmu, kritik dan saran.
6. Seluruh civitas academika Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Informatika Universitas Brawijaya dan selama penyelesaian skripsi ini.
7. Serta semua pihak yang telah membantu secara langsung maupun tidak langsung sehingga penulis dapat menyelesaikan skripsi ini.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 2 Januari 2019

Penulis
Email: try.rahayuni5@gmail.com

ABSTRAK

Tri Rahayuni, Pencarian Resep Makanan Berdasarkan Citra Makanan Menggunakan Ekstraksi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*

Pembimbing: Yuita Arum Sari, S.Kom., M.Kom dan Sigit Adinugroho, S.Kom., M.Sc.

Aplikasi pencarian resep makanan yang ada hanya menggunakan *query* teks. Penggunaan *query* teks seringkali tidak mewakili semua yang diinginkan oleh pengguna dan tidak bisa dilakukan jika pegguna hanya mengetahui gambar makanan. Solusi yang ditawarkan untuk mengatasi hal tersebut adalah dibuat pencarian resep makanan dengan menggunakan citra makanan. Pencarian citra dilakukan dengan mengukur kemiripan antara fitur citra *query* dengan fitur citra *corpus*. Fitur-fitur pada citra didapatkan dengan ekstraksi fitur *Simple Morphological Shape Descriptors* dan *Color Moment*. Setelah ekstraksi fitur selanjutnya dilakukan pengukuran kemiripan menggunakan *Euclidean Distance*. Sistem kemudian akan menampilkan hasil pencarian yaitu sebanyak n citra yang memiliki tingkat kemiripan terbesar. Hasil penelitian ini menunjukkan nilai MAP tertinggi pada *k-rank* 10 yaitu 95,713% dan nilai MAP terendah pada *k-rank* 100 yaitu 76,108%. Penggunaan fitur *Color Moment* lebih baik dari pada *Simple Morphological Shape Descriptors* karena nilai MAP *Color Moment* lebih tinggi yaitu 93,32% dari pada *Simple Morphological Shape Descriptors* yaitu 89,8%. Penggabungan kedua fitur terbukti mampu meningkatkan nilai MAP menjadi 95,71%. Dapat disimpulkan bahwa pada *k-rank* 10 sistem mengembalikan hasil yang baik sesuai kebutuhan pengguna dan penggunaan gabungan kedua fitur dapat mengatasi kelemahan dari penggunaan masing-masing fitur.

Kata kunci: CBIR, Resep Makanan, Citra Makanan, *Simple Morphological Shape Descriptors*, *Color Moment*

ABSTRACT

Tri Rahayuni, Food Recipe Search Based on Food Images Using Extraction Simple Morphological Shape Descriptors and Color Moment Features
Advisor: Yuita Arum Sari, S.Kom., M.Kom and Sigit Adinugroho, S.Kom., M.Sc.

The existing food recipe search application only uses text queries. The use of text queries often does not represent everything the user wants and cannot be done if the user only knows food images. The solution offered to overcome this problem is to make a food recipe search using food imagery. Image search is done by measuring the similarity between query image features and corpus image features. The features of the image are obtained by extracting the Simple Morphological Shape Descriptors and Color Moment features. After feature extraction the measurements of similarity were then performed using Euclidean Distance. The system will then display the search results which are as many as n images that have the greatest degree of similarity. The results of this study indicate the highest MAP value at k-rank 10 is 95.713% and the lowest MAP value is at k-rank 100 which is 76.108%. The use of the Color Moment feature is better than the Simple Morphological Shape Descriptors because the MAP Color Moment value is higher at 93.32% than the Simple Morphological Shape Descriptors which is 89.8%. The merging of the two features proved to be able to increase the MAP value to 95.71%. It can be concluded that at k-rank 10 the system returns good results according to user requirements and the use of the two merged features can overcome the disadvantages of using each feature.

Keywords: CBIR, Food Recipes, Food Images, Simple Morphological Shape Descriptors, Color Moment

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Seiring dengan berkembangnya zaman, variasi makanan juga ikut berkembang. Perkembangan tersebut dapat dilihat dari semakin banyaknya jenis makanan baru yang bermunculan baik karya asli maupun hasil modifikasi makanan yang sudah ada. Namun tidak semua orang tahu bagaimana cara membuat setiap jenis makanan. Sehingga dibutuhkan resep makanan sebagai alat bantu pembuatan makanan. Masyarakat biasaya membeli buku resep untuk mendapatkan resep makanan. Pencarian pada buku resep sering kali menyulitkan karena harus membuka perlembar halaman sehingga membutuhkan waktu lama. Oleh sebab itu, diperlukan adanya aplikasi yang memudahkan masyarakat untuk mencari resep makanan (Irawati & Sugiarti, 2016).

Resep makanan tidak terlepas dari citra makanan. Hampir semua pencarian resep makanan diikuti dengan hasil eksplorasi citra makanan. Bahkan semakin menarik citra makanan yang ditampilkan, akan semakin menarik minat masyarakat untuk mencoba resep tersebut. Selama ini aplikasi pencarian resep makanan seperti Cookpad hanya menggunakan *query* teks judul atau nama makanan tanpa melibatkan citra makanan. Aplikasi pencarian resep makanan lainnya yang sudah dikembangkan menggunakan pencarian berdasarkan bahan baku, judul masakan, dan jenis masakan yang diinginkan pengguna (Lestari & Kursini, 2015). Proses pencarian dengan menggunakan *query* teks bisa jadi tidak mewakili semua yang diinginkan oleh pengguna (Agaputra, et al., 2013). Selain itu pencarian dengan *query* teks tidak bisa dilakukan jika pegguna hanya mengetahui gambar dari sesuatu yang dicari (Frediansah, et al., 2012). Pengguna akan kesulitan melakukan pencarian jika hanya mengetahui gambar makanan yang ingin dicari resepnya. Solusi yang ditawarkan untuk mengatasi masalah tersebut adalah dengan menerapkan sistem temu kembali citra pada pencarian resep makanan.

Pada sistem temu kembali citra, pencarian citra dilakukan dengan mencocokkan fitur-fitur pada citra (Frediansah, et al., 2012). Pencocokan dilakukan dengan cara mengukur kemiripan antara fitur citra *query* dengan fitur citra pada *corpus*. Citra yang memiliki tingkat kemiripan tinggi dengan citra masukan akan ditampilkan sebagai hasil keluaran. Fitur-fitur pada citra didapatkan dengan melakukan ekstraksi fitur pada citra. Jenis fitur yang biasa digunakan yaitu warna, tekstur, dan bentuk (Wäldchen & Mäder, 2018). Salah satu metode ekstraksi fitur bentuk yaitu *Simple Morphological Shape Descriptors*. Metode tersebut pernah digunakan oleh Caglayan, et al. (2016) untuk mengklasifikasikan citra daun. Percobaan dengan menggunakan ekstraksi bentuk *Simple Morphological Shape Descriptors* menghasilkan akurasi yang cukup baik yaitu 87,61% (Caglayan, et al., 2016).

Hasil dari beberapa penelitian yang sudah dilakukan menunjukkan bahwa metode *Simple Morphological Shape Descriptors* kurang memadai untuk membedakan citra yang memiliki perbedaan kecil (Wäldchen & Mäder, 2018). Sedangkan pada citra makanan seringkali memiliki bentuk yang hampir serupa

misalnya bentuk roti tawar dengan roti gandum yang sama-sama berbentuk persegi. Oleh sebab itu pada penelitian ini metode tersebut dikombinasikan dengan *Color Moment* untuk ekstraksi warna. Penelitian yang mengimplementasikan *Color Moment* pernah dilakukan oleh Kusuma, et al. (2017). Pada penelitian tersebut akurasi terbaik diperoleh ketika menggunakan fitur warna *Color Moment* yaitu sebesar 87,00% (Kusuma, et al., 2017). *Color Moment* memiliki dimensi dan kompleksitas komputasi yang rendah sehingga cocok untuk aplikasi *real-time* (Wäldchen & Mäder, 2018). Penelitian yang menggabungkan metode *Simple Morphological Shape Descriptors* dan *Color Moment* pernah dilakukan dan terbukti dapat meningkatkan akurasi klasifikasi citra daun secara signifikan dari 87,61% menjadi 96% (Caglayan, et al., 2016).

Pengukuran kemiripan digunakan untuk membandingkan citra *query* dengan citra pada *corpus*. Penelitian yang membandingkan beberapa metode pengukuran kemiripan pernah dilakukan oleh Nugraheny pada tahun 2015. Penelitian tersebut membandingkan metode *Euclidean Distance*, *Manhattan City Block Distance* dan *Mahalonobis Distance* untuk mengukur kemiripan citra awan pada sistem deteksi awan *cumulonimbus*. Hasilnya metode *Euclidean Distance* menghasilkan akurasi terbaik yaitu 93%, kemudian diikuti metode *Manhattan City Block Distance* dengan akurasi 90%, dan metode *Mahalanobis* dengan akurasi terendah yaitu 50% (Nugraheny, 2015). Oleh sebab itu, pada penelitian ini menerapkan *Euclidean Distance* sebagai metode untuk pengukuran kemiripan.

Berdasarkan paparan tersebut, diusulkan penelitian dengan judul "Pencarian Resep Makanan Berdasarkan Citra Makanan Menggunakan Ekstraksi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*".

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dijelaskan, berikut rumusan masalah pada penelitian ini.

1. Bagaimana akurasi pencarian resep makanan berdasarkan citra makanan menggunakan ekstraksi fitur *Simple Morphological Shape Descriptors* dan *Color Moment*?
2. Bagaimana pengaruh ekstraksi fitur *Simple Morphological Shape Descriptors* dan *Color Moment* terhadap pencarian resep makanan berdasarkan citra makanan?

1.3 Tujuan

Berikut merupakan tujuan yang ingin dicapai pada penelitian ini.

1. Menguji akurasi pencarian resep makanan berdasarkan citra makanan menggunakan ekstraksi fitur *Simple Morphological Shape Descriptors* dan *Color Moment.*
2. Mengetahui pengaruh ekstraksi fitur *Simple Morphological Shape Descriptors* dan *Color Moment* terhadap pencarian resep makanan berdasarkan citra makanan.

1.4 Manfaat

Penelitian ini diharapkan dapat memberikan manfaat sebagai berikut.

1. Melalui penelitian ini dapat diketahui akurasi pencarian resep makanan berdasarkan citra makanan menggunakan ekstraksi fitur *Simple Morphological Shape Descriptors* dan *Color Moment*.
2. Melalui penelitian ini dapat diketahui pengaruh ekstraksi fitur *Simple Morphological Shape Descriptors* dan *Color Moment* terhadap pencarian resep makanan berdasarkan citra makanan.

1.5 Batasan masalah

Berikut merupakan batasan masalah pada penelitian ini.

1. Ekstraksi bentuk menggunakan metode *Simple Morphological Shape Descriptors* yang meliputi *major axis length*, *minor axis length*, diameter, *centroid*, area, *perimeter*, *aspect ratio*, *roundness*, *rectangularity*, *compactness, narrow factor*, rasio *perimeter* dengan *diameter,* rasio *perimeter* dengan *major axis length,* rasio *perimeter* dengan *major axis length* dan *minor axis length.*
2. Ekstraksi warna menggunakan *color space* RGB.
3. Kemiripan citra diukur dengan menggunakan metode *Euclidean Distance*.
4. Format citra yang digunakan .jpg.
5. Citra yang digunakan terbatas pada citra makanan tunggal.
6. Citra makanan yang digunakan terbatas pada makanan padat.
7. Citra makanan yang digunakan terbatas pada makanan yang tidak memiliki warna sama dengan *backrgound*.

1.6 Sistematika Pembahasan

Sistematika penyusunan laporan pada penelitian ini meliputi beberapa bab, dengan rincian sebagai berikut.

BAB I PENDAHULUAN

Bab ini berisi latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah, dan sistematika pembahasan pada Pencarian Resep Makanan Berdasarkan Citra Makanan Menggunakan Ekstraksi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*.

BAB II LANDASAN KEPUSTAKAAN

Bab ini berisi kajian pustaka dan dasar teori yang digunakan untuk mendukung penelitian Pencarian Resep Makanan Berdasarkan Citra Makanan Menggunakan Ekstraksi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*.

BAB III METODOLOGI PENELITIAN

Metodologi penelitian berisi tahapan-tahapan yang ada pada proses perancangan Pencarian Resep Makanan Berdasarkan Citra Makanan Menggunakan Ekstraksi Fitur *Simple Morphological Shape Descriptors dan Color Moment* serta langkah kerja yang dilakukan dalam penelitian.

BAB IV PERANCANGAN

Pada bab ini berisi tentang perancangan Pencarian Resep Makanan Berdasarkan Citra Makanan Menggunakan Ekstraksi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*.

BAB V IMPLEMENTASI

Bab ini membahas tentang implementasi Ekstraksi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment* pada Pencarian Resep Makanan Berdasarkan Citra Makanan.

BAB VI PENGUJIAN DAN ANALISIS

Bab keenam ini membahas pengujian pada tingkat akurasi pada Pencarian Resep Makanan Berdasarkan Citra Makanan Menggunakan Ekstraksi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*.

BAB VII PENUTUP

Bab ini menguraikan kesimpulan yang diperoleh dari pengujian Pencarian Resep Makanan Berdasarkan Citra Makanan Menggunakan Ekstraksi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment* dan menambahkan saran yang digunakan untu pengembangan selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini terdiri dari dua bagian yaitu kajian pustaka dan dasar teori. Pada bagian kajian pustaka membahas secara umum penelitian-penelitian terdahulu yang berhubungan dengan penelitian yang diusulkan. Sedangkan pada dasar teori membahas teori-teori yang diperlukan untuk menyusun penelitian yang diusulkan. Dasar teori yang akan dibahas pada bab ini meliputi resep makanan, citra digital, *content base image retrieval*, ekstraksi fitur yang meliputi *Simple Morphological Shape Descriptors* dan *Color Moment*, normalisasi, *Euclidean Distance* untuk mengukur kemiripan dan evaluasi.

2.1 Kajian Pustaka

Penelitian yang menggunakan *Simple Morphological Shape Descriptors* pernah dilakukan oleh Aakif dan Faisal. Penelitian tersebut digunakan untuk mengklasifikasikan pohon buah menggunakan citra daun. Pada penelitian tersebut metode *Simple Morphological Shape Descriptors* dikombinasikan dengan metode lain yaitu *Fourier Descriptors* dan *Shape Defining Features* yang sama-sama digunakan sebagai ekstraksi fitur bentuk. Fitur *Simple Morphological Shape Descriptors* yang digunakan meliputi *aspect ratio*, *eccentricity*, *roundness* dan *convex hull*. Akurasi tertinggi didapat dari kombinasi seluruh ekstraksi fitur yang menghasilkan akurasi 96,5% (Aakif & Faisal, 2015).

Beberapa penelitian yang menggunakan *Color Moment* sebagai ekstraksi fitur warna pernah dilakukan. Penelitian yang dilakukan oleh Sarker dan Iqbal menggabungkan ekstraksi fitur warna yaitu *Color Moment* dan fitur tekstur yaitu *Haar Wavelet Transform*. *Color Moment* yang digunakan yaitu *mean*, *standard deviasion* dan *skewness*. Pada proses pengujian dilakukan tiga kali percobaan yaitu dengan fitur tekstur saja, fitur warna saja dan kombinasi fitur tekstur dan warna. Akurasi terbaik didapat pada percobaan dengan menggabungkan fitur tekstur dan warna yaitu 88,0%. Sedangkan percobaan dengan mengimplementasikan fitur warna saja menghasilkan akurasi 77,0% (Sarker & Iqbal, 2013).

Penelitian yang dilakukan oleh Kaipravan tahun 2016 juga mengimplementasikan metode *Color Moment*. Pada penelitian tersebut *Color Moment* dikombinasikan dengan *Gabor Filter* untuk ekstraksi fitur tekstur. Sedangkan untuk mengukur kemiripan antar citra digunakan *Manhattan distance*. *Color Moment* yang digunakan yaitu mean dan standard deviasi. Hasil dari penelitian tersebut menyimpulkan bahwa gabungan ekstraksi fitur *Color Moment* dan *Gabor Filter* dapat meningkatkan efisiensi *image retrieval* (Kaipravan, Muhsina; Rejiram, R, 2016).

Penelitian lain juga mengimplementasikan Color Moment sebagai ekstraksi fitur. Penelitian tersebut mengkombinasikan metode *Color Moment* dan *Local Binary Pattern* sebagai ekstraksi fitur tekstur. *Euclidean distance* digunakan untuk mengukur kemiripan antar citra *query* dengan citra *corpus*. *Color Moment* yang

digunakan meliputi mean, standard deviasi dan skewness. Dari 10 kali percobaan dengan membedakan jumlah gambar serta menggunakan metode LBP, CM dan LBP dan CM memberikan hasil yang beragam. Rata-rata nilai precision terbaik didapatkan pada percobaan menggunakan metode CM dan gabungan LBP dan CM dengan precision tertinggi sebesar 100% (Singh & Agrawal, 2017).

Penelitian lain yang menggunakan *Color Moment* sebagai salah satu ekstraksi fiturnya pernah dilakukan oleh Kusuma, dkk. Penelitian tersebut mengkombinasikan *Color Moment* dan *Gray Level Co-occurence Matrix* (GLCM). Pada proses klasifikasi, penelitian tersebut melakukan percobaan terhadap dua metode diantaranya *K-Nearest Neighbor* dan *Support Vector Machine*. *Color moment* yang digunakan pada penelitian tersebut yaitu *skewness*, *standard deviation* dan *mean*. Klasifikasi paling baik diperoleh saat menggunakan fitur Color Moment dengan algoritme SVM pada kernel linier yaitu 87.00% (Kusuma, et al., 2017).

Penelitian-penelitian yang menggabungkan metode *Simple Morphological Shape Descriptors* dan *Color Moment* pernah dilakukan. Penelitian pertama dilakukan oleh Caglayan, dkk. Penelitian tersebut menggunakan dua belas fitur yang meliputi *smoothness* daun, aspek rasio, *form factor*, *rectangularity*, *narrow factor,* rasio perimeter dan panjang, rasio perimeter dengan panjang dan lebar serta lima struktur morphological. Hasil penelitian tersebut menghasilkan bahwa algoritme *Random Forest* dengan ekstraksi gabungan fitur bentuk dan warna menghasilkan akurasi terbaik yaitu sebesar 96% (Caglayan, et al., 2016). Penelitian kedua dilakukan oleh Prasad, dkk. Fitur yang digunakan pada metode *Simple Morphological Shape Descriptors* meliputi elongation, roundness, circularity dan porosity. Sedangkan fitur yang digunakan pada *Color Moment* meliputi *mean*, *standard deviation*, *skewness* dan *kurtosis*. Hasil penelitian membuktikan bahwa kombinasi ekstraksi fitur *Simple Morphological Shape Descriptors* dan *Color Moment* menghasilkan akurasi tertinggi yaitu sebesar 91,34% (Prasad, et al., 2013).

2.2 Resep Makanan

Resep makanan merupakan suatu petunjuk yang memuat nama makanan, bahan, bumbu dan teknik yang digunakan dalam mengolah bahan makanan (Lestari & Kursini, 2015). Resep makanan berfungsi untuk mempermudah orang dalam membuat makanan. Saat ini resep makanan sudah menjamur di internet yang dapat diakses oleh siapapun. Dengan memasukkan *query* nama makanan yang dimaksud search engine akan menampilkan banyak resep makanan. Pencarian resep makanan saat ini hanya sebatas menggunakan *query* teks.

2.3 Content Base Information Retrieval

Content Base Image Retrieval (CBIR) merupakan aplikasi pengolahan citra yang digunakan untuk mendapatkan atau mencari secara cepat citra yang ada di basis data berdasarkan permintaan *user* (Azis, 2013). Untuk mendapatkan citra yang ada di basis data diperlukan *query* citra sebagai masukan dari pengguna. Sistem akan mengubah *query* citra masukan tersebut menjadi bentuk vektor fitur

yang akan dibandingkan tingkat kemiripannya dengan vektor fitur basis data. Kemudian dilakukan proses pengurutan berdasarkan tingkat kemiripan tinggi sampai rendah. Sistem kemudian akan menampilkan hasil pencarian citra yaitu citra yang memiliki tingkat kemiripan terbesar sebanyak n citra.

2.4 Citra Digital

Citra digital merupakan representasi citra yang ditangkap oleh mesin dengan bentuk pendekatan berdasarkan sampling dan kuantisasi (Primahayu, et al., 2017). Sampling menyatakan besar kotak-kotak yang tersusun dalam bentuk baris dan kolom. Sampling menyatakan ukuran piksel pada citra dan kuantisasi citra menyatakan jumlah warna yang ada pada citra.

2.4.1 *Gaussian Filter*

Gaussian filter merupakan salah satu metode *filtering* yang mana menggunakan fungsi *gaussian* untuk melakukan pembobotan untuk setiap anggotanya (Afifa, 2016). *Gaussian filter* berfungsi untuk mereduksi *noise* yang bersifat sebaran normal. Untuk menghitung nilai setiap elemen dalam *gaussian filter* menggunakan rumus pada Persamaan 2.1 (Sutarno, et al., 2017).

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.1}$$

Keterangan:
G(x,y) : nilai elemen gaussian pada posisi (x,y)
π : konstanta dengan nilai 3.14
e : konstanta dengan nilai 2.71828182846
σ : *standard deviation*

2.4.2 Konversi RGB ke L*a*b

Pada color spaces L*a*b, komponen L* berhubungan dengan *brightness*, komponen a* untuk *red-green* dan b* untuk *yellow-blue*. Konversi nilai RGB ke bentuk L*a*b menggunakan rumus pada Persamaan 2.2. Nilai masing-masing komponen L*, a*, dan b* ditunjukkan oleh Persamaan 2.3 sampai 2.5 (Rejito, 2013).

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0.490 & 0.310 & 0.200 \\ 0.177 & 0.813 & 0.011 \\ 0.000 & 0.010 & 0.990 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{2.2}$$

$$L^* = 25\left(\frac{100}{Y_0}\right)^{\frac{1}{3}} - 16 \tag{2.3}$$

$$a^* = 500\left[\left(\frac{X}{X_0}\right)^{\frac{1}{3}} - \left(\frac{Y}{Y_0}\right)^{\frac{1}{3}}\right] \tag{2.4}$$

$$b^* = 200\left[\left(\frac{Y}{Y_0}\right)^{\frac{1}{3}} - \left(\frac{Z}{Z_0}\right)^{\frac{1}{3}}\right] \quad (2.5)$$

2.4.3 *Otsu Thresholding*

Thresholding merupakan proses mengubah citra menjadi citra biner dengan menggunakan nilai *threshold* (T) untuk membedakan daerah objek dengan daerah yang termasuk *background* (Ambarwati, et al., 2017). Jika nilai piksel lebih besar dari *threshold* maka nilai piksel menjadi 1 dan jika sebaliknya diatur menjadi 0. Salah satu metode *thresholding* adalah *Otsu Thresholding*. Langkah-langkah pada *otsu thresholding* adalah sebagai berikut (Syafi'i, et al., 2015).

1. Membuat histogram untuk mengetahui jumlah piksel pada setiap tingkat warna. Setiap tingkat warna pada citra dinyatakan dengan level *i* sampai *L* yang mana level *i* adalah level ke-1 dengan nilai piksel 0 dan level *L* adalah level ke-256 dengan nilai piksel 255. Nilai *threshold* untuk suatu citra *grayscale* dinyatakan dengan *k* dengan kisaran nilai 0 sampai *L-1*. Probabilitas setiap piksel pada level *i* dinyataan dengan Persamaan 2.6.

$$P_i = \frac{n_i}{N} \quad (2.6)$$

Keterangan:
P_i : Probabilitas piksel ke-*i*
n_i : Banyaknya piksel dengan tingkat warna *i*
N : Total banyaknya piksel pada citra

2. Mencari nilai jumlah kumulatif, rerata kumulatif dan intensitas global. Rumus untuk masing-masing nilai tersebut dapat dilihat pada Persamaan 2.7 sampai 2.9.

$$\omega(k) = \sum_{i=0}^{k} P_i \quad (2.7)$$

$$\mu(k) = \sum_{i=0}^{k} i.P_i \quad (2.8)$$

$$\mu_T(k) = \sum_{i=0}^{k} i.P_i \quad (2.9)$$

Keterangan:
$\omega(k)$: jumlah kumulatif ke-k
P_i : Probabilitas piksel ke-*i*
$\mu(k)$: rerata kumulatif
$\mu_T(k)$: intensitas global

3. Menentukan varian antar kelas dengan menggunakan persamaan 2.10. Hasil perhitungan varian antar kelas dicari nilai maksimal dan digunakan sebagai threshold menggunakan Persamaan 2.11.

$$\sigma_B^2(k) = \frac{[\mu_T \omega(k) - \mu(k)]^2}{\omega(k)[1 - \omega(k)]} \quad (2.10)$$

$$\sigma_B^2(k*) = max_{1 \le x \le L} \sigma_B^2(k) \quad (2.11)$$

2.4.4 Operasi Morfologi

Morfologi merupakan serangkaian piksel-piksel yang membentuk kumpulan data dua dimensi pada citra digital (Setiawan, et al., 2016). Operasi morfologi bertujuan untuk mengubah bentuk objek pada citra menjadi lebih tebal atau lebih tipis. Operasi morfologi berfungsi memudahkan mengenali objek pada citra dengan cara meningkatkan aspek bentuk dan struktur. Pada operasi morfologi masukan yang digunakan yaitu citra biner dan kernel (Primahayu, et al., 2017). Kernel atau *structuring element* (SE) merupakan elemen pembentuk struktur yang berupa matriks dan umumnya berukuran kecil. Contoh kernel ditunjukkan pada Gambar 2.1. Terdapat dua operasi yang mendasari operasi morfologi yaitu erosi dan dilasi (Primahayu, et al., 2017).

1	1	1
1	1	1
1	1	1

Gambar 2.1 Contoh Kernel

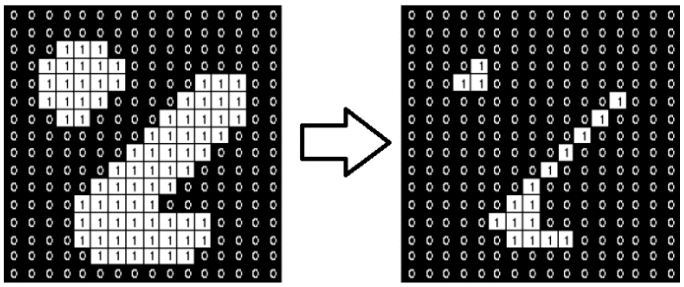
2.4.4.1 Erosi

Operasi erosi digunakan untuk menghapus atau mengurasi piksel-piksel suatu objek citra sehingga ukuran objek citra akan terlihat lebih kecil (Setiawan, et al., 2016). Operasi erosi digunakan untuk menghilangkan objek-objek kecil yang tidak berguna untuk proses selanjutnya. Piksel yang dihapus adalah lapisan piksel terluar suatu objek citra. Contoh operasi erosi dapat dilihat pada Gambar 2.2. Rumus operasi dilasi ditunjukkan oleh Persamaan 2.12 (Setiawan, et al., 2016).

$$A \ominus B = \{Z | (B)z \subseteq A\} \quad (2.12)$$

Persamaan 2.12 menjelaskan bahwa erosi terjadi pada citra A dengan menggunakan kernal B terdiri atas semua titik z = (x, y) yang mana (B)z ada di dalam himpunan A. Kemudian B digeser sedemikian hingga di dalam A tepat pada tepinya dan dicari pada bagian mana saja B bebar-benar ada di dalam A. Untuk

kondisi yang memenuhi syarat tersebut maka area yang bersesuaian dengan B perlu ditandai.



Gambar 2.2 Operasi Erosi

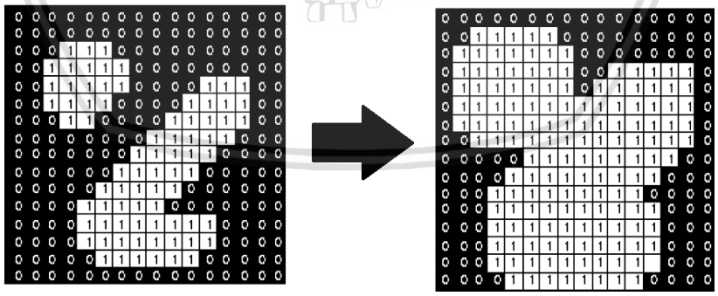
Sumber: (devtrik.com)

2.4.4.2 Dilasi

Operasi dilasi adalah kebalikan dari operasi erosi. Operasi dilasi digunakan untuk menebalkan objek pada citra. Proses penebalan objek dilakukan dengan cara menggabungkan titik piksel latar menjadi bagian suatu objek sesuai dengan kernel yang digunakan (Primahayu, et al., 2017). Dilasi berfungsi untuk membuat objek pada citra terlihat lebih jelas dengan cara menambal bagian kecil citra yang tidak terhubung. Operasi dilasi dapat dilihat pada Gambar 2.3. Rumus operasi dilasi ditunjukkan pada Persamaan 2.13 (Setiawan, et al., 2016).

$$A \oplus B = \{Z|(B)z \subseteq A \neq \emptyset\} \quad (2.13)$$

Ini berarti bahwa untuk setiap area di luar tepi citra A akan dilakukan translasi atau pergeseran dan kemudian menggabungkan seluruh hasilnya (union) dengan hasil translasi strel B.



Gambar 2.3 Operasi Dilasi

Sumber: (devtrik.com)

2.4.5 *Cropping*

Cropping adalah proses memotong bagian citra untuk mendapatkan citra yang lebih terfokus pada objek citra yang akan diproses (Sandy, et al., 2018). Selain itu proses *cropping* juga akan lebih menyederhanakan ukuran citra. Proses *cropping* dapat dilakukan dengan mencari nilai koordinat titik maksimum dan

minimum dari koordinat citra (Sandy, et al., 2018). Citra yang digunakan sebagai masukan proses *cropping* adalah citra biner atau citra hitam putih. Contoh yang sebelum dilakukan proses *cropping* dan setelah dilakukan proses *cropping* ditunjukkan pada Gambar 2.4.

Gambar 2.4 Contoh *Cropping*

2.4.6 *Canny Edge Detector*

Canny edge detector adalah salah satu metode deteksi tepi yang ditemukan oleh John F. Canny. Langkah-langkah deteksi tepi canny adalah sebagai berikut (Mazid, 2013).

1. *Image smoothing*
   Pada langkah ini berfungsi untuk menghilangkan *noise* dengan cara mengaburkan gambar.
2. *Finding gradien*
   Langkah selanjutnya yaitu mencari gradien terbesar kemudian menandai tepian pada gambar tersebut.
3. *Non maximum suppression*
   Bagian ini berfungsi untuk menghasilkan tepian tajam dari tepian yang blured hasil finding gradien dengan cara mengkonversikan tepian tersebut.
4. *Double thresholding*
   Double thresholding yaitu tepian yang berpotensi ditentukan oleh *thresholding*.
5. *Edge tracking by hysteresis*
   Kemudian menentukan tepian final dengan cara menekan semua sisi yang tidak terhubung dengan tepian yang sangat kuat.

2.5 Ekstraksi Fitur

Ekstraksi fitur merupakan proses untuk mendapatkan fitur atau ciri-ciri yang terdapat pada sebuah citra. Ekstraksi fitur terdiri atas peta vektor pengamatan yang meliputi struktur berbasis metode untuk mendeteksi struktur citra seperti tepi, garis, sudut, lingkaran, elips, dan lain lain (Setiawan, et al., 2016). Hasil dari ekstraksi fitur dapat digunakan sebagai representasi citra yang menyamakan atau membedakan antara citra makanan yang satu dengan lainnya. Beberapa proses ekstraksi fitur mungkin membutuhkan untuk mengubah citra menjadi citra biner,

citra *grayscale* atau bentuk citra lainnya. Fitur yang dapat di ekstrak dari sebuah citra makanan meliputi fitur bentuk dan fitur warna. Ekstraksi fitur bentuk menggunakan metode *Simple Morphological Shape Descriptors*. Sedangkan ekstraksi fitur warna menggunakan metode *Color Moment*.

2.5.1 *Simple Morphological Shape Descriptors*

Terdapat enam fitur dasar pada *Simple Morphological Shape Descriptors* yang digunakan untuk analisis bentuk objek yang meliputi *major axis length*, diameter*, minor axis length*, *perimeter,* area, dan *centroid* (Wäldchen & Mäder, 2018). Beberapa penelitian sering menggunakan rasio sebagai deskriptor bentuk. Rasio dipilih karena memiliki perhitungan yang sederhana dan secara alami invarian untuk *translation*, *rotation* dan *scaling*, membuat rasio kuat terhadap representasi berbeda dari objek yang sama (Wäldchen & Mäder, 2018). *Aspect ratio*, r*oundness*, *rectangularity, compactness, narrow factor,* rasio *perimeter* dengan *diameter,* rasio *perimeter* dengan *major axis length* dan rasio *perimeter* dengan *major axis length* dan *minor axis length* adalah fitur rasio yang dapat digunakan. Citra yang digunakan pada ekstraksi fitur *Simple Morphological Shape Descriptors* adalah citra biner. Penjelasan untuk fitur-fitur tersebut adalah sebagai berikut.

a. *Major axis length* (L) adalah panjang garis yang menghubungkan antara dasar dan ujung objek.
b. *Minor axis length* (W) adalah lebar maksimum yang tegak lurus dengan major axis.
c. Diameter (D) didefinisikan sebagai jarak terpanjang antara dua titik pada tipe objek.
d. *Centroid* merupakan respresentasi titik koordinat yang terletak pada tengah objek.
e. Area (A) merupakan perhitungan jumlah piksel pada daerah objek.
f. *Perimeter* (P) merupakan perhitungan jumlah piksel pada tepi objek.
g. *Aspect ratio* (AR) merupakan rasio perbandingan antara *major axis length* dengan *minor axis length*. Fitur ini digunakan untuk memperkirakan bentuk objek makanan. Jika bernilai lebih dari 1 maka objek tersebut melebar. Jika kurang dari 1 maka bentuk objek tersebut memanjang. Persamaan untuk menghitung *aspect ratio* dapat dilihat pada Persamaan 2.14.

$$AR = \frac{L}{W} \tag{2.14}$$

h. *Roundness* (R) mengilustrasikan perbedaan antara objek dan lingkaran. Fitur ini digunakan untuk mengukur seberap bundar bentuk objek makanan tersebut. Persamaan untuk menghitung *roundness* dapat dilihat pada Persamaan 2.15.

$$R = \frac{4\pi A}{p^2} \tag{2.15}$$

i. *Rectangularity* (N) mengilustrasikan kemiripan antara objek dengan empat persegi panjang. Persamaan untuk menghitung *rectangularity* dapat dilihat pada Persamaan 2.16.

$$N = \frac{A}{LW} \tag{2.16}$$

j. *Compactness* (C) merupakan rasio antara perimeter dengan area objek, fitur ini memberikan informasi tentang kompleksitas umum dan faktor bentuk. Persamaan untuk menghitung *compactness* dapat dilihat pada Persamaan 2.17.

$$C = \frac{P}{\sqrt{A}} \tag{2.17}$$

k. *Narrow Factor* (NF) merupakan rasio antara diameter dengan major axis length. Persamaan untuk menghitung *narrow factor* dapat dilihat pada Persamaan 2.18.

$$NF = \frac{D}{L} \tag{2.18}$$

l. Rasio *perimeter* dengan diameter (P_D) merupakan ukuran perbandingan perimeter dengan diameter. Persamaan untuk menghitung rasio perimeter dengan diameterdapat dilihat pada Persamaan 2.19.

$$P_D = \frac{P}{D} \tag{2.19}$$

m. Rasio *perimeter* dengan *major axis length* (P_L) merupakan ukuran perbandingan perimeter dengan *major axis length*. Persamaan untuk menghitung Rasio *Perimeter* dengan *Major Axis Length* dapat dilihat pada Persamaan 2.20.

$$P_L = \frac{P}{L} \tag{2.20}$$

n. Rasio *Perimeter* dengan *Major axis length* dan *Minor axis length* (P_{LW}) merupakan rasio perimeter dengan jumlah *major axis length* dan *minor axis length*. Persamaan untuk menghitung Rasio *Perimeter* dengan *Major axis length* dan *Minor axis length* dapat dilihat pada Persamaan 2.21.

$$P_{LW} = \frac{P}{(L+W)} \tag{2.21}$$

2.5.2 *Color Moment*

Color moment merupakan salah satu fitur warna yang digunakan sebagai karakterisasi suatu citra. Perhitungan *color moment* digunakan untuk mendapatkan kesamaan *query* citra makanan dengan *corpus* citra. *Color Moment* mengasumsikan distribusi warna sebuah gambar sebagai distribusi probabilitas. Tiga moment warna yang terbukti efektif dan efisien digunakan untuk mewakili distribusi warna adalah *mean, standard deviation* dan *skewness* (Singh & Agrawal, 2017). Penelitian lain menambahkan *kurtosis* sebagai fitur color moment (Sutarno, et al., 2017). *Mean* adalah nilai rata-rata piksel warna warna pada citra (Sutarno, et al., 2017). *Standard deviasi* diperoleh dengan cara mengambil akar kuadrat dari

variasi distribusi warna (Sarker & Iqbal, 2013). *Skewness* merupakan derajat ketidaksimetrisan suatu distribusi dan dikatakan simetris ketika bernilai nol sebaliknya dikatakan tidak simetris jika bernilai 1 (Sutarno, et al., 2017). *Kurtosis* adalah derajat keruncingan suatu distribusi warna yang diukur relatif terhadap distribusi normal (Sutarno, et al., 2017). *Color moment* distribusi tersebut dihitung untuk setiap *channel Red, Green dan Blue*. Sehingga *Color Moment* menghasilkan 9 fitur. Rumus untuk menghitung *mean*, standard deviasi, *skewness, kurtosis* masing-masing didefinisikan pada Persamaan 2.22, 2.23, 2.24 dan 2.25 (Sutarno, et al., 2017).

$$\mu_k = \frac{1}{M.N}\sum_{i=1}^{M}\sum_{j=1}^{N} x_{ij} \tag{2.22}$$

$$\sigma_k = \sqrt{\frac{1}{M.N}\sum_{i=1}^{M}\sum_{j=1}^{N}(x_{ij}-\mu_k)^2} \tag{2.23}$$

$$\theta_k = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(x_{ij}-\mu_k)^3}{MN\sigma_k^3} \tag{2.24}$$

$$\gamma_k = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}(x_{ij}-\mu_k)^4}{MN\sigma_k^4} \tag{2.25}$$

Keterangan.
μ_k = rata-rata untuk masing masing channel ke-k (R,G,B)
M = panjang citra
N = lebar citra
x_{ij} = nilai piksel pada citra panjang ke-i lebar ke-j
σ_k = standart deviasi untuk setiap *channel* k
θ_k= skewness untuk setiap *channel* k
γ_k = kurtosis untuk setiap *channel* k

2.6 Normalisasi Data

Normalisasi data bertujuan untuk membuat data memiliki rentang nilai yang sama. Hal ini dikarenakan data yang bernilai lebih besar menyebabkan data yang lebih kecil menjadi terkesampingkan (Basheer & Hajmeer, 2000). Rentang nilai yang umum digunakan pada normalisasi data adalah rentang 0 sampai 1. Salah satu metode yang digunakan untuk normalisasi data adalam *min max normalization*. Metode *min max normalization* akan me-*rescale* data dari suatu rentang ke rentang baru dengan skala 0 sampai 1 (Chamidah, et al., 2012). Rumus untuk menghitung *min max normalization* dapat dilihat pada Persamaan 2.26 (Chamidah, et al., 2012).

$$s' = \frac{s - \min\{sk]}{\max\{sk\} - \min\{sk\}} \tag{2.26}$$

Keterangan:

s' : data hasil normalisasi

s : data asli
$min\{s_k\}$: nilai minimum data asli
$max\{s_k\}$: nilai maksimum data asli

2.7 Euclidean Distance

Pengukuran kemiripan (*Similarity Measure*) digunakan untuk mengukur kemiripan antara *query* citra dengan citra *corpus*. Pengukuran kemiripan yang digunakan pada penelitian ini adalah *Euclidean Distance*. *Euclidean distance* merupakan pengembangan dari metode *Minkowski* dengan Lp-Norm dengan nilai p=2. Pengukuran *Euclidean distance* menggunakan perhitungan akar dari kuadrat dua atau lebih perbedaan vektor (Amynarto, et al., 2018). Persamaan *Euclidean distance* dijelaskan pada Persamaan 2.27 (Singh & Agrawal, 2017).

$$di = \sqrt{\sum_{i=1}^{p} (x_{2i} - x_{1i})^2} \quad (2.27)$$

Yang mana x_1 dan x_2 merupakan vektor *query* citra dan vektor citra di basis data sebagai hasil dari ekstraksi fitur bentuk dan warna. Semakin kecil nilai *di* maka semakin dekat vektor *query* citra dengan vektor citra basis data yang dibandingkan.

2.8 Evaluasi

Pada temu kembali citra, sistem mengembalikan sekumpulan citra sebagai hasil pencarian citra oleh pengguna. Terdapat dua kategori citra yang dihasilkan yaitu *relevant* citra dan *retrieved* citra. *Relevant* citra adalah citra yang *relevant* dengan query citra yang dicari oleh pengguna sedangkan *retrieved* citra adalah citra yang diterima sebagai hasil pengembalian pencarian. Evaluasi dapat digunakan meliputi *ranked retrieval* dan *unranked retrieval*. Evaluasi *ranked retrieval* yang dapat digunakan yaitu *Mean Average Precission* (MAP) (Sari, et al., 2014). Evaluasi MAP merupakan hasil perhitungan rata-rata dokumen relevan yang *retrieved* dari setiap *query* yang terlibat di dalam sistem (Baskoro, et al., 2015). Nilai MAP mempunyai rentang nilai 0 sampai 1, dan dalam sebuah system dikatakan baik jika nilai MAP mendekati 1. Rumus dari MAP dapat dilihat pada Persamaan 2.28 (Baskoro, et al., 2015).

$$MAP = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m} \sum_{k=1}^{m_j} precision(R_{jk}) \quad (2.28)$$

Keterangan:
MAP : nilai *Mean Average Precission*
Q : banyaknya *query* yang diinputkan
R_{jk} : nilai *precission* yang di-*retrieve*
m : banyaknya citra yang di-*retrieve*

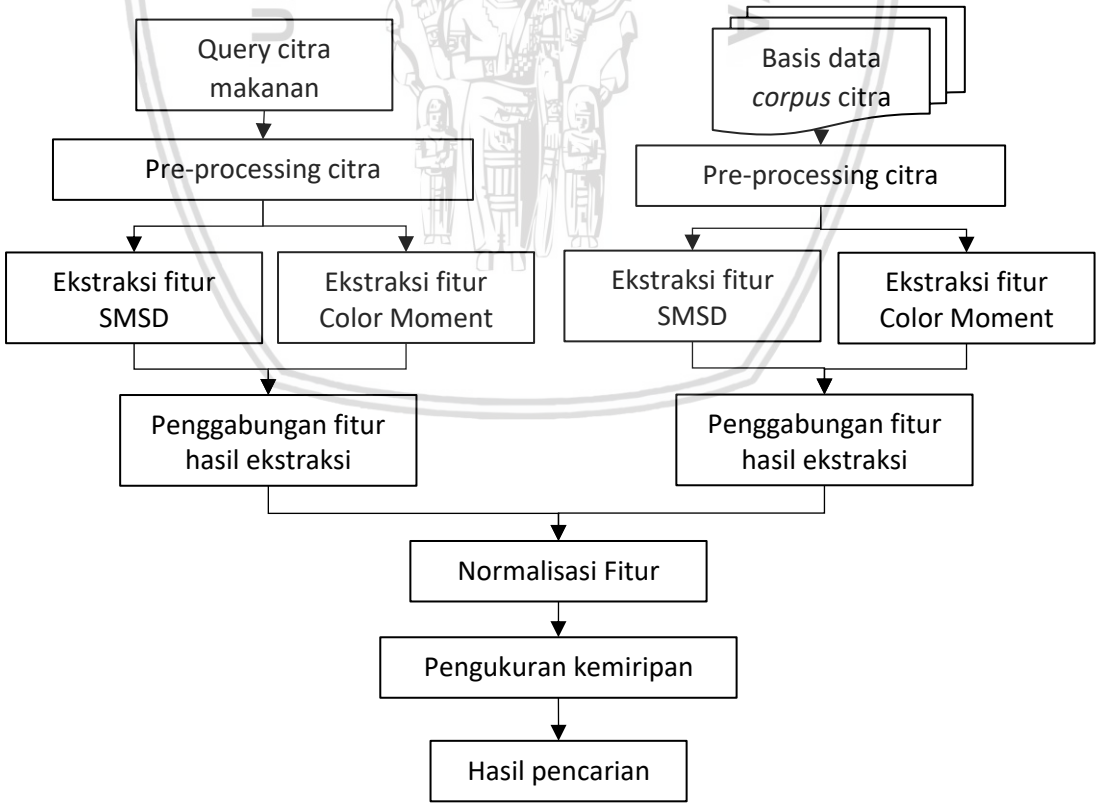
BAB 3 METODOLOGI PENELITIAN

3.1 Tipe Penelitian

Tipe penelitian ini termasuk ke dalam penelitian non implementatif yang mana penelitian ini menitikberatkan pada analisis terhadap hubungan antar fenomena yang sedang dikaji. Ditinjau dari kegiatan penelitiannya, pendekatan pada penelitian ini bertipe analitik yang menjelaskan derajat hubungan antar elemen dalam objek penelitian.

3.2 Metode Penelitian

Metode penelitian dijelaskan pada Gambar 3.1 Sistem menerima masukan berupa *query* citra makanan yang akan dicari yang selanjutnya akan dilakukan *pre-processing* citra. Proses selanjutnya yaitu ekstraksi fitur *Simple Morphological Shape Descriptors* dan fitur *Color Moment*. Kemudian hasil kedua ekstraksi fitur tersebut digabungkan menjadi sebuah vektor. Langkah yang sama juga dilakukan untuk semua *corpus* citra yang ada di basis data. Setelah itu dilakukan pengukuran kemiripan dengan menggunakan *Euclidean Distance*. Hasil proses tersebut diurutkan dan diambil sejumlah *n* tertinggi dan akan ditampilkan sebagai *output* pencarian.

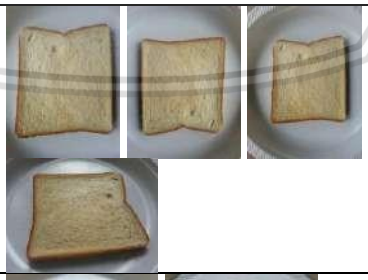
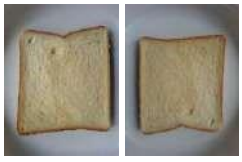


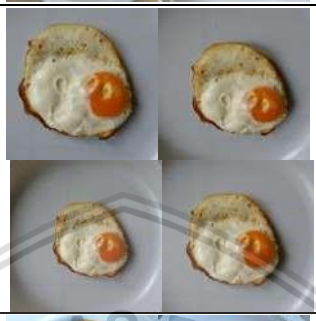
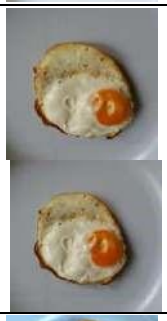
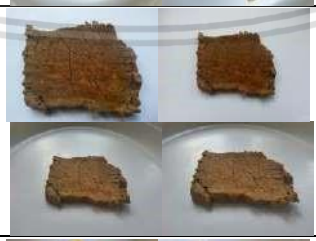
Gambar 3.1 Metode Penelitian

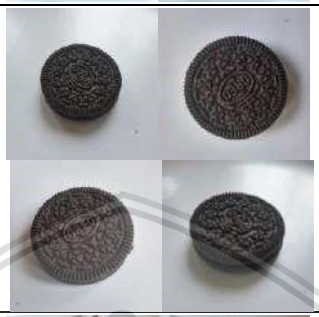
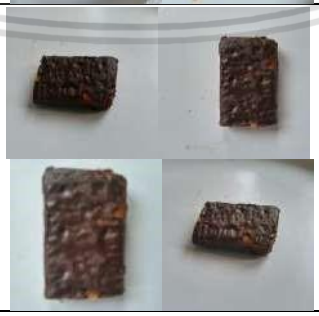
3.3 Objek Penelitian

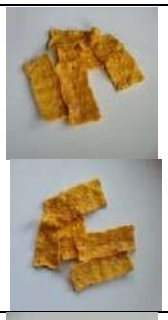
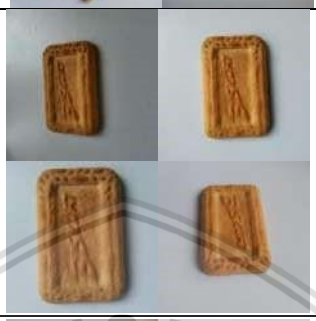
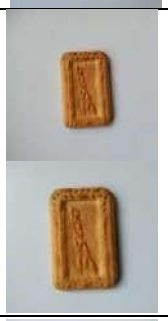
Objek yang digunakan pada penelitian ini adalah citra makanan tunggal. Data terdiri dari 19 jenis makanan yang meliputi Donat, Roti gandum, Roti tawar, Mie keriting, Mie gepeng, Telor ceplok, Telor dadar, *Fried chicken*, Rendang, Nasi kuning, Nasi merah, Oreo, Beng-beng, Mie kering, Wafer coklat, Keripik kentang, Biskuit coklat, Bola-bola coklat, dan Gabin. Setiap jenis makanan berjumlah 10 yang dibagi menjadi 8 citra sebagai *corpus* dan 2 citra sebagai *query*. Jadi total terdapat 190 citra makanan yang digunakan sebagai objek penelitian. Citra makanan tersebut disimpan dalam format jpg. Tabel 3.1 adalah sampel citra makanan yang digunakan sebagai objek pada penelitian ini.

Tabel 3.1 Sampel Citra Makanan yang Digunakan

Nama Makanan	Kode	Data Training	Data Testing
Donat	001		
Roti Gandum	002		
Roti tawar	003		
Mie keriting	004		

Mie gepeng	005		
Telur ceplok	006		
Telur dadar	007		
Fried chicken	008		
Rendang	009		
Nasi kuning	021		

Nasi merah	022		
Oreo	023		
Beng-beng	024		
Mie kering	025		
Wafer coklat	026		

Keripik kentang	027		
Biskuit coklat	029		
Bola-bola coklat	030		
Gabin	031		

3.4 Lokasi Penelitian

Lokasi penelitian bertempat di Laboratorium Riset Komputasi Cerdas, Ruang F9.3, Gedung F, Fakultas Ilmu Komputer, Universitas Brawijaya.

3.5 Metode Pengumpulan Data

Data citra makanan pada penelitian ini didapatkan dengan cara memotret makanan secara langsung dengan menggunakan kamera *smartphone*. Sedangkan data resep makanan didapat dari aplikasi Cookpad. Detail pengumpulan data citra makanan adalah sebagai berikut.

a. Pengambilan citra makanan dilakukan dengan menggunakan *smartphone* LG Stylus 2 dengan kamera utama 13 Megapiksel.

b. Pencahayaan yang digunakan saat memotret citra makanan adalah cahaya matahari pada Ruang Grup Riset *Computer Vision*, Gedung F Fakultas Ilmu Komputer Lantai 9.
c. Pengambilan citra makanan dilakukan secara tegak lurus dengan ketinggian yang berbeda dan dengan kemiringan tertentu.

3.6 Analisis Kebutuhan

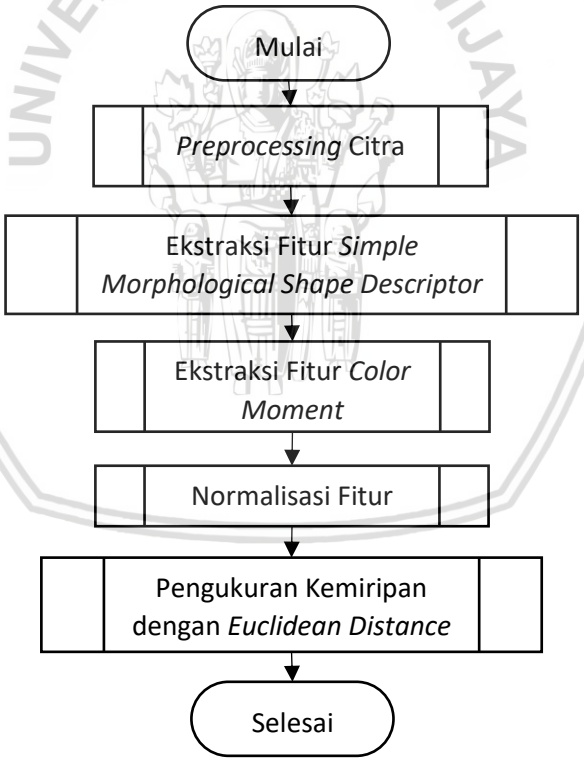
Analisis kebutuhan bertujuan untuk menganalisis dan mendapatkan semua kebutuhan yang diperlukan untuk pembuatan "Pencarian Resep Makanan Berdasarkan Citra Makanan Menggunakan Ekstraksi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*". Analisis kebutuhan disesuaikan dengan kebutuhan *hardware*, kebutuhan *software* dan kebutuhan fungsional. Berikut ini kebutuhan yang digunakan:

1. Kebutuhan *hardware*
   a. Processor AMD Quad Core X4
   b. RAM 4 GB
   c. Harddisk dengan kapasitas 500 GB
2. Kebutuhan *software*
   a. Sistem Operasi Windows 10
   b. Python 2
   c. JetBrains Pycharm Community Edition 2017.2.2.
   d. Library Python : Open CV, Numpy
3. Kebutuhan fungsional
   Sistem dapat menampilkan hasil pencarian makanan sesuai denga citra *query*.

BAB 4 PERANCANGAN

4.1 Perancangan Sistem

Perancangan sistem pencarian resep makanan berdasarkan citra makanan dilakukan dengan mengimplementasikan ekstraksi fitur bentuk yaitu *Simple Morphological Shape Descriptors* dan ekstraksi fitur warna yaitu *Color Moment*. Proses perhitungan kemiripan menggunakan metode *Euclidean Distance*. Sebelum dilakukan proses perhitungan kemiripan, citra harus dilakukan *pre-processing* kemudian dilanjutkan dengan ekstraksi fitur citra. *Pre-processing* berfungsi untuk mengolah citra sehingga mudah untuk mengekstrak fiturnya. Fitur citra didapatkan dengan cara melakukan ekstraksi fitur pada citra makanan. Kemudian dilakukan normalisasi fitur menggunakan *min-max normalization*. Kemudian dilakukan pengukuran kemiripan antara citra *query* dengan *corpus* citra yang ada di *database*. Langkah-langkah proses pencarian resep makanan berdasarkan citra makanan dapat dilihat pada Gambar 4.1.

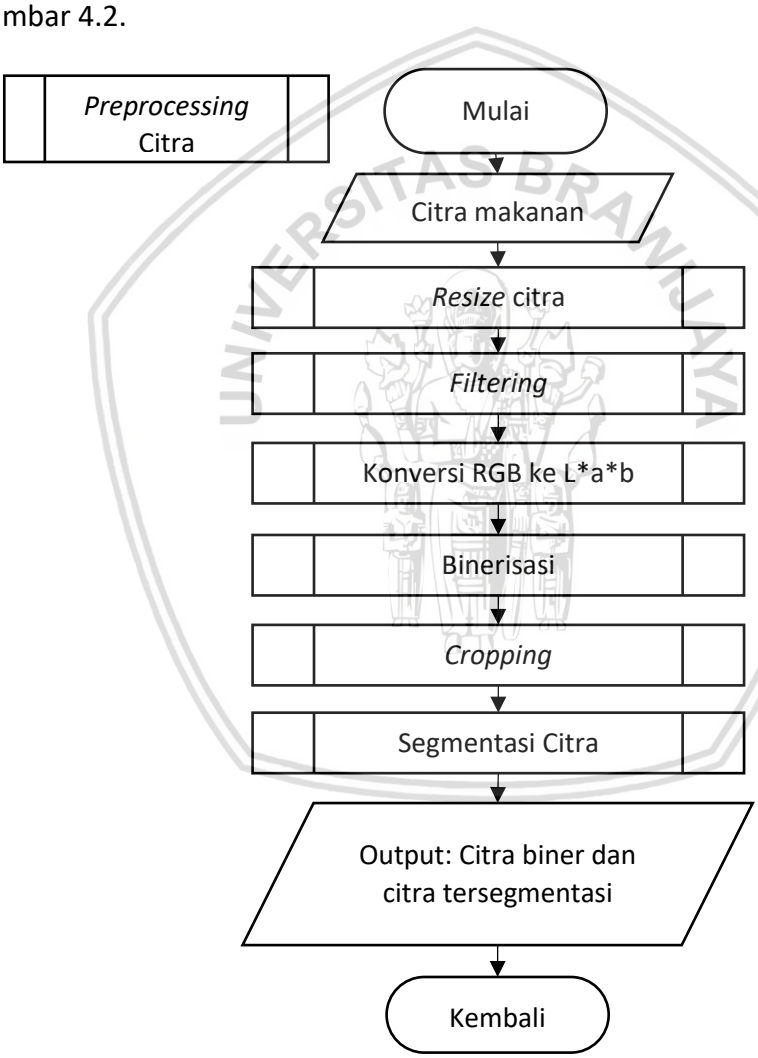


Gambar 4.1 Diagram Alir Perancangan Sistem

4.1.1 *Pre-processing* Citra

Tahap ini berfungsi untuk mempersiapkan citra menuju proses selanjutnya yaitu ekstraksi fitur. *Pre-processing* terdiri dari beberapa tahapan yaitu *resize* citra, *filtering*, konversi RGB ke L*a*b, binerisasi, *cropping* serta segmentasi citra. Tahap

resize citra merupakan tahap untuk menyeragamkan ukuran citra dengan cara mengubah ukuran citra sehingga memiliki panjang 500 piksel dan lebar 500 piksel. Selanjutnya tahap *filtering* yang berfungsi untuk menghilangkan *noise* pada citra. Kemudian dilakukan perubahan *color space* dari RGB ke L*a*b untuk mendapatkan komponen b citra L*a*b. Kemudian dilakukan tahap binerisasi yaitu mengubah komponen B citra L*a*b menjadi citra biner. Pada citra biner akan tampak lebih jelas antara bagian objek dan bagian *background*. Setelah itu dilakukan *cropping* untuk mendapatkan citra baru yang hanya berisi objek makanan. Tahap selanjutnya yaitu segementasi citra yang menghasilkan citra dengan *background* hitam. *Pre-processing* menghasilkan dua citra keluaran yaitu citra biner dan citra tersegmentasi. Diagram alir *pre-processing* dapat dilihat pada Gambar 4.2.

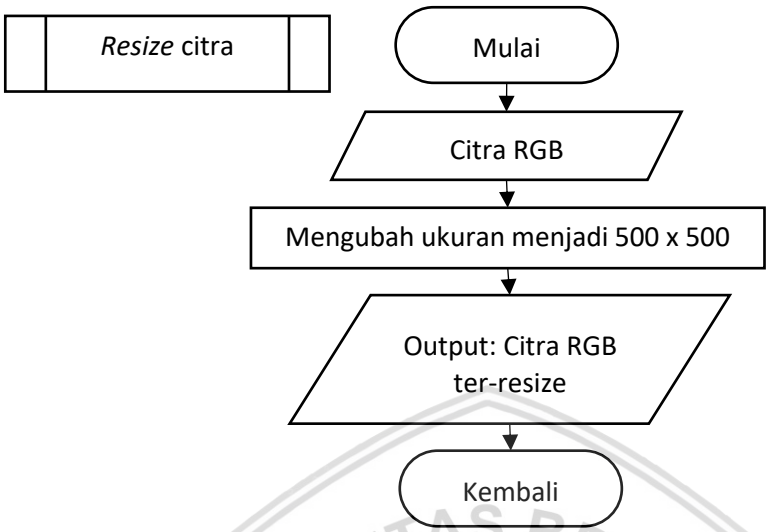


Gambar 4.2 Diagram Alir *Pre-processing* Citra

4.1.1.1 *Resize Citra*

Setiap citra dilakukan proses *resize* citra yaitu proses untuk mengubah ukuran citra menjadi 500 x 500 piksel. Sehingga citra-citra yang awalnya memiliki

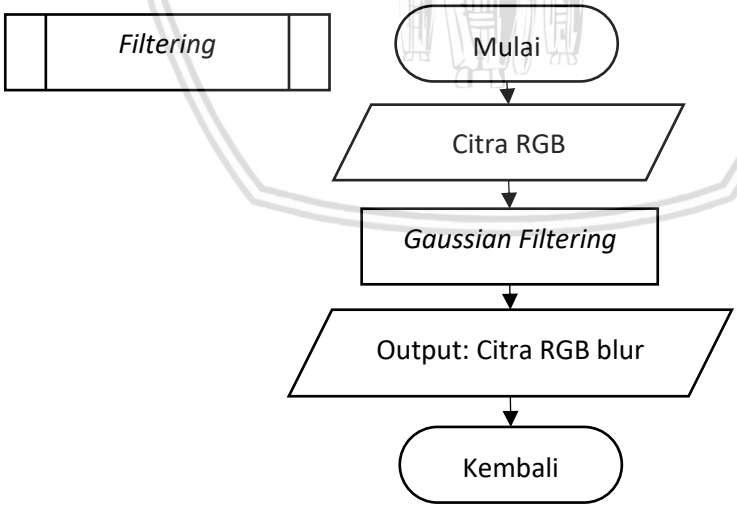
ukuran berbeda-beda bisa memiliki ukuran yang sama. Proses *resize* citra dapat dilihat pada Gambar 4.3.



Gambar 4.3 Diagram Alir *Resize* Citra

4.1.1.2 *Filtering*

Setiap citra makanan dilakukan *filtering* untuk menghilangkan *noise*. Proses *filtering* berguna untuk tahap memisahkan objek makanan dengan background. Masukan prooses filtering adalah citra RGB. Proses *filtering* pada penelitian ini menggunakan *Gaussian Filtering*. Keluaran porses filtering adalah citra RGB blur. Proses *filtering* dapat dilihat pada Gambar 4.4.

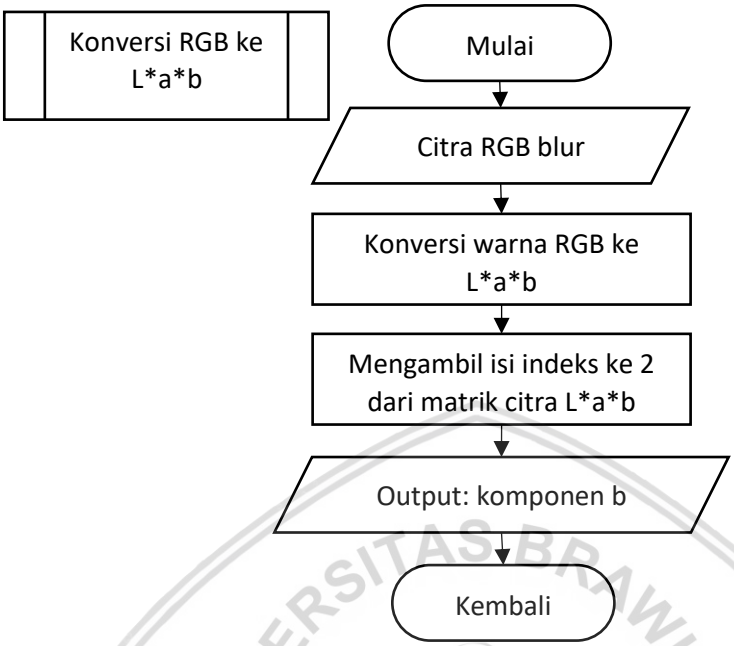


Gambar 4.4 Diagram Alir *Filtering*

4.1.1.3 Konversi RGB ke L*a*b

Tahap ini dimulai dengan mengubah citra RGB blur menjadi citra dengan *color space* L*a*b. Kemudian dilakukan ekstraksi komponen b dari citra L*a*b.

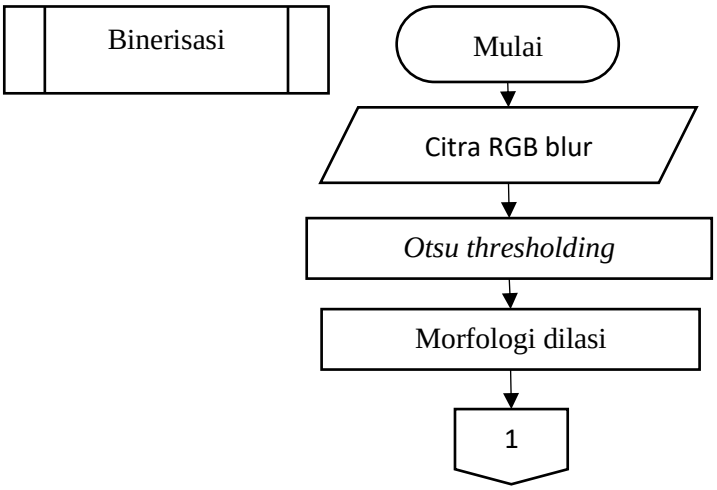
Hasil akhir dari proses ini adalah matriks komponen b citra L*a*b. Gambar 4.5 menunjukkan langkah-langkah dari pengubahan citra RGB ke L*a*b.

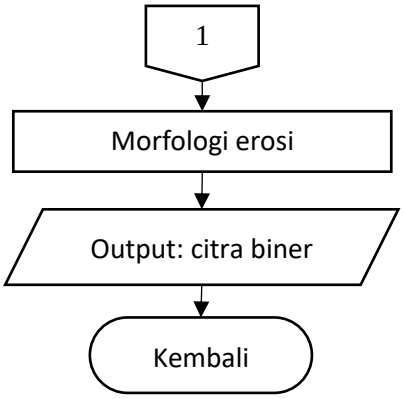


Gambar 4.5 Diagram Alir Konversi RGB ke L*a*b

4.1.1.4 Binerisasi

Binerisasi merupakan proses untuk mengubah citra menjadi citra biner atau hitam putih. Komponen b citra L*a*b yang menjadi masukan dilakukan *thresholding* menggunakan *Otsu Thresholding*. Proses *thresholding* menghasilkan citra hitam putih yang mana objek makanan berwarna putih dan *background* berwarna hitam. Setelah itu dilakukan operasi morfologi *closing* yang terdiri dari dilasi dan erosi. Tahap ini berfungsi mengisi lubang kecil pada objek dan menghaluskan batas dari objek tanpa mengubah ukuran objek makanan tersebut. Hasil akhir proses binerisasi adalah citra biner. Secara keseluruhan tahap binerisasi dapat dilihat pada Gambar 4.6.

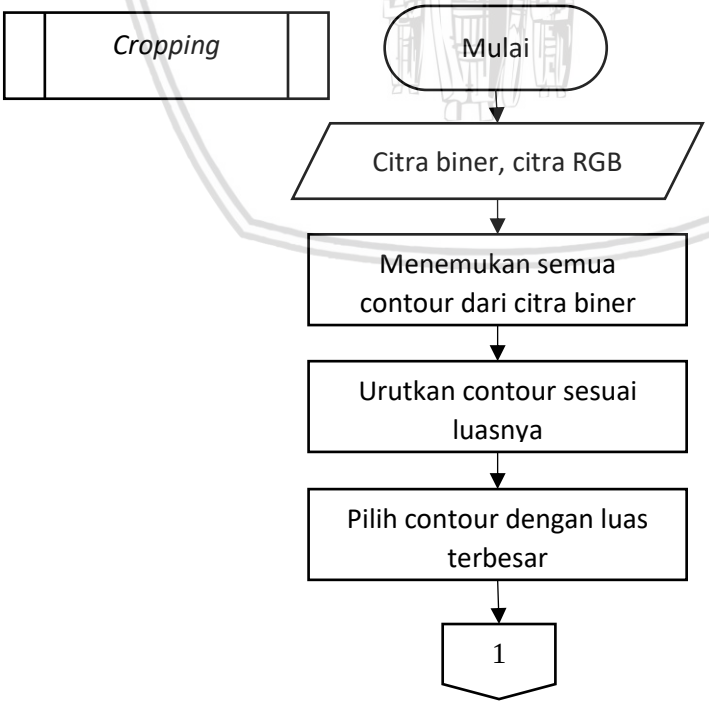


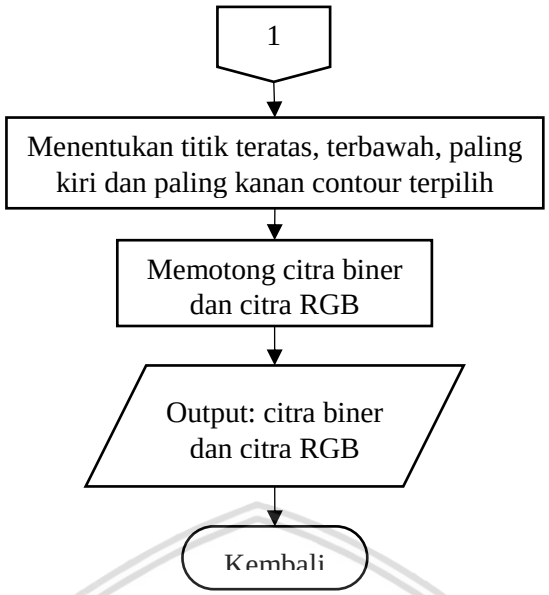


Gambar 4.6 Diagram Alir Binerisasi

4.1.1.5 *Cropping*

Cropping dilakukan untuk mendapatkan piksel-piksel yang berisi objek makanan sehingga citra yang diolah pada tahap berikutnya yaitu citra yang berisi objek saja. Masukan proses *cropping* adalah citra biner. Proses *cropping* dimulai dengan menemukan semua *contour* pada citra biner kemudian menghitung luas dari masing – masing *contour* tersebut. Hasil perhitungan luas kemudian diurutkan dari luas terbesar sampai terkecil. Kemudian dipilih *contour* terbesar untuk dihitung koordinat teratas, terbawah, paling kiri dan paling kanan. Kemudian dilakukan pemotongan sesuai dengan titik-titik tersebut. Hasil akhir yaitu citra RGB dan citra biner yang terpotong. Proses *cropping* dapat dilihat pada Gambar 4.7.

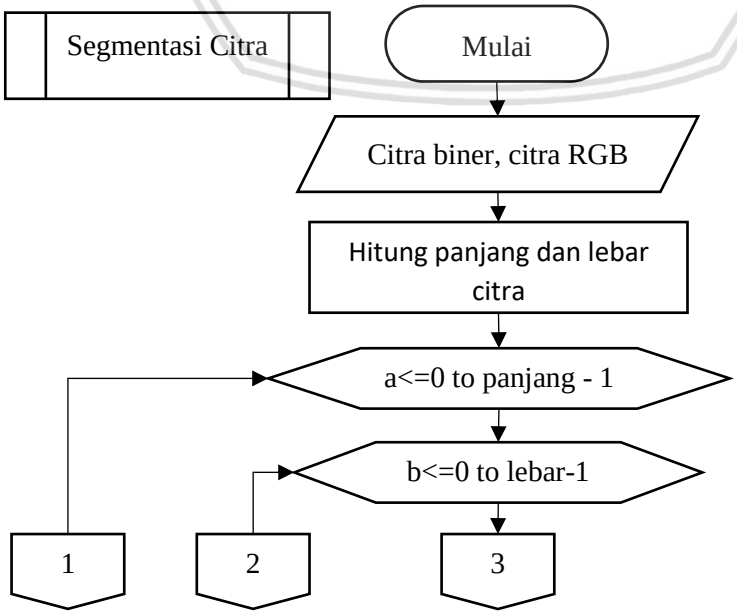


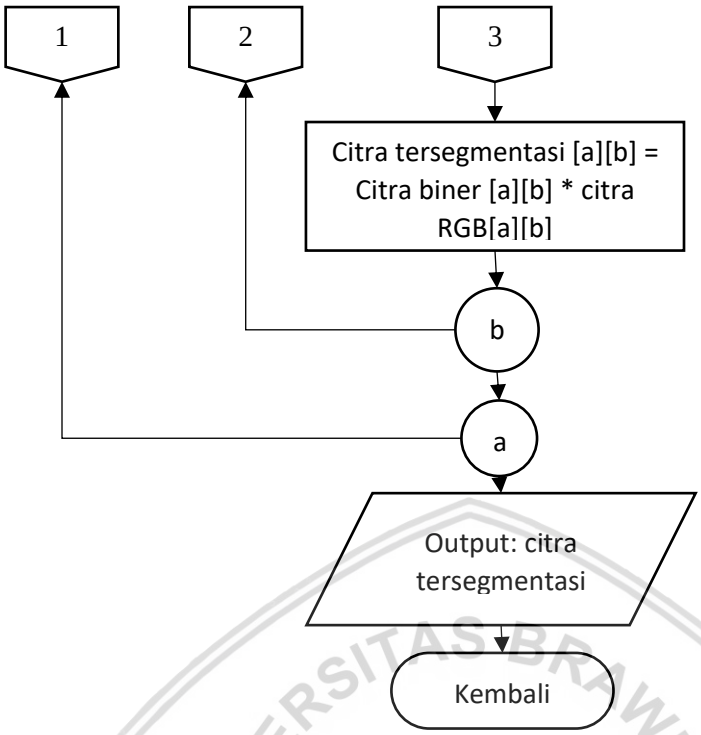


Gambar 4.7 Diagram Alir *Cropping*

4.1.1.6 Segmentasi Citra

Setelah melalui proses-proses sebelumnya, proses terakhir dari *pre-processing* yaitu segmentasi citra. Segmentasi citra berfungsi untuk memisahkan *background* dengan objek makanan. Proses ini akan menghasilkan citra tersegmentasi yaitu citra RGB yang memiliki *background* berwarna hitam. Masukan proses segmentasi adalah citra RGB dan citra biner yang terpotong. Proses segmentasi citra dimulai dengan menghitung panjang dan lebar citra kemudian dilakukan perulangan untuk mengalikan setiap piksel citra biner dengan piksel citra RGB. Hasil proses segmentasi adalah citra tersegmentasi yaitu citra RGB yang memiliki *background* hitam. Proses segmentasi citra dapat dilihat pada Gambar 4.8.

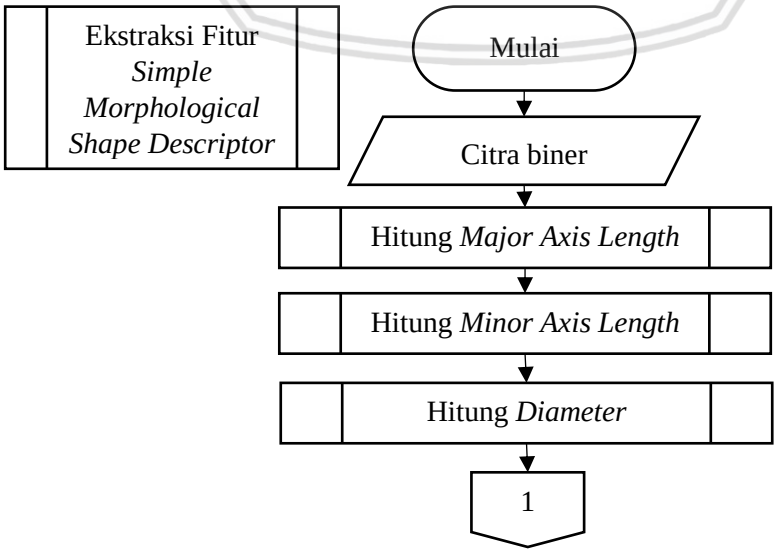


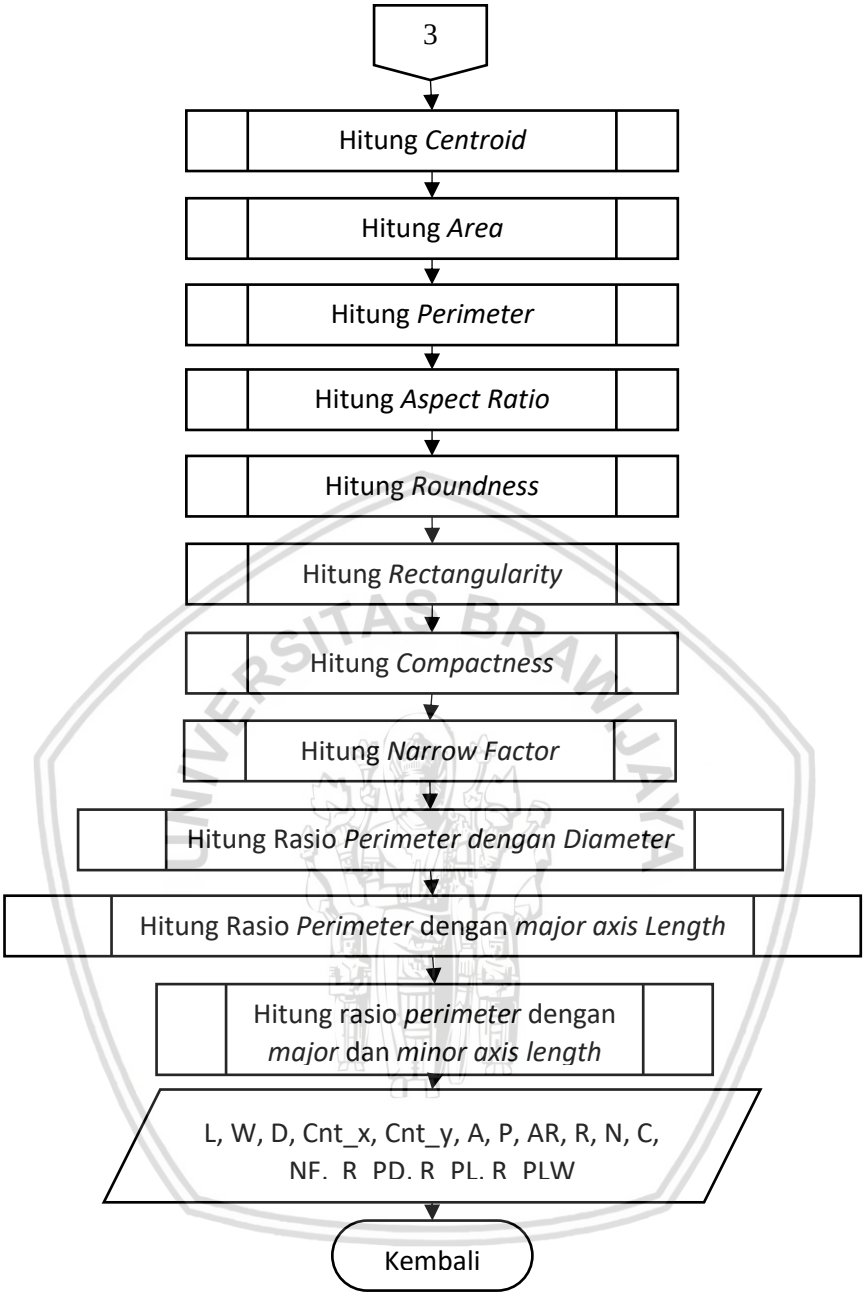


Gambar 4.8 Diagram Alir Segmentasi

4.1.2 Ekstraksi Fitur *Simple Morphological Shape Descriptors*

Tahap ekstraksi fitur *Simple Morphological Shape Descriptors* merupakan tahap pengambilan fitur bentuk yang terdiri dari lima belas fitur. Semua fitur tersebut diantaranya *major axis length*, *minor axis length*, diameter, *centroid x, centroid y, area*, *perimeter*, *aspect ratio*, *roundness*, *rectangularity*, *compactness*, *narrow factor*, rasio *perimeter* dengan *diameter,* rasio *perimeter* dengan *major axis length,* rasio *perimeter* dengan *major axis length* dan *minor axis length*. Diagram alir proses ekstraksi fitur *Simple Morphological Shape Descriptors* dapat dilihat pada Gambar 4.9.

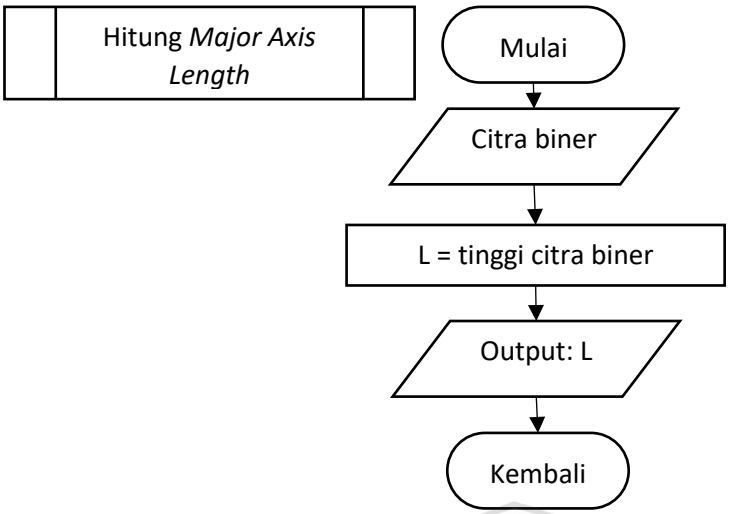




Gambar 4.9 Diagram Alir Ekstraksi Fitur *Simple Morphological Shape Descriptors*

4.1.2.1 *Major Axis Length*

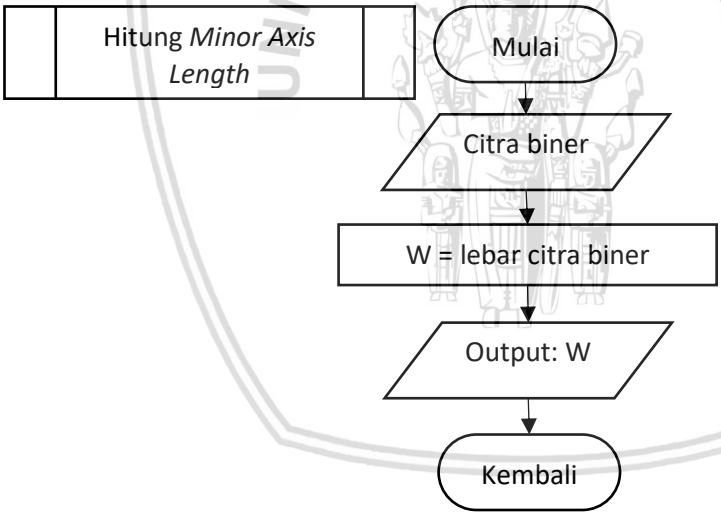
Fitur *major axis leng*th didapatkan dengan mengukur tinggi objek makanan. Citra masukan yang digunakan adalah citra biner. Gambar 4.10 menunjukkan proses untuk mendapatkan fitur *major axis length*.



Gambar 4.10 Diagram Alir Hitung *Major Axis Length*

4.1.2.2 *Minor Axis Length*

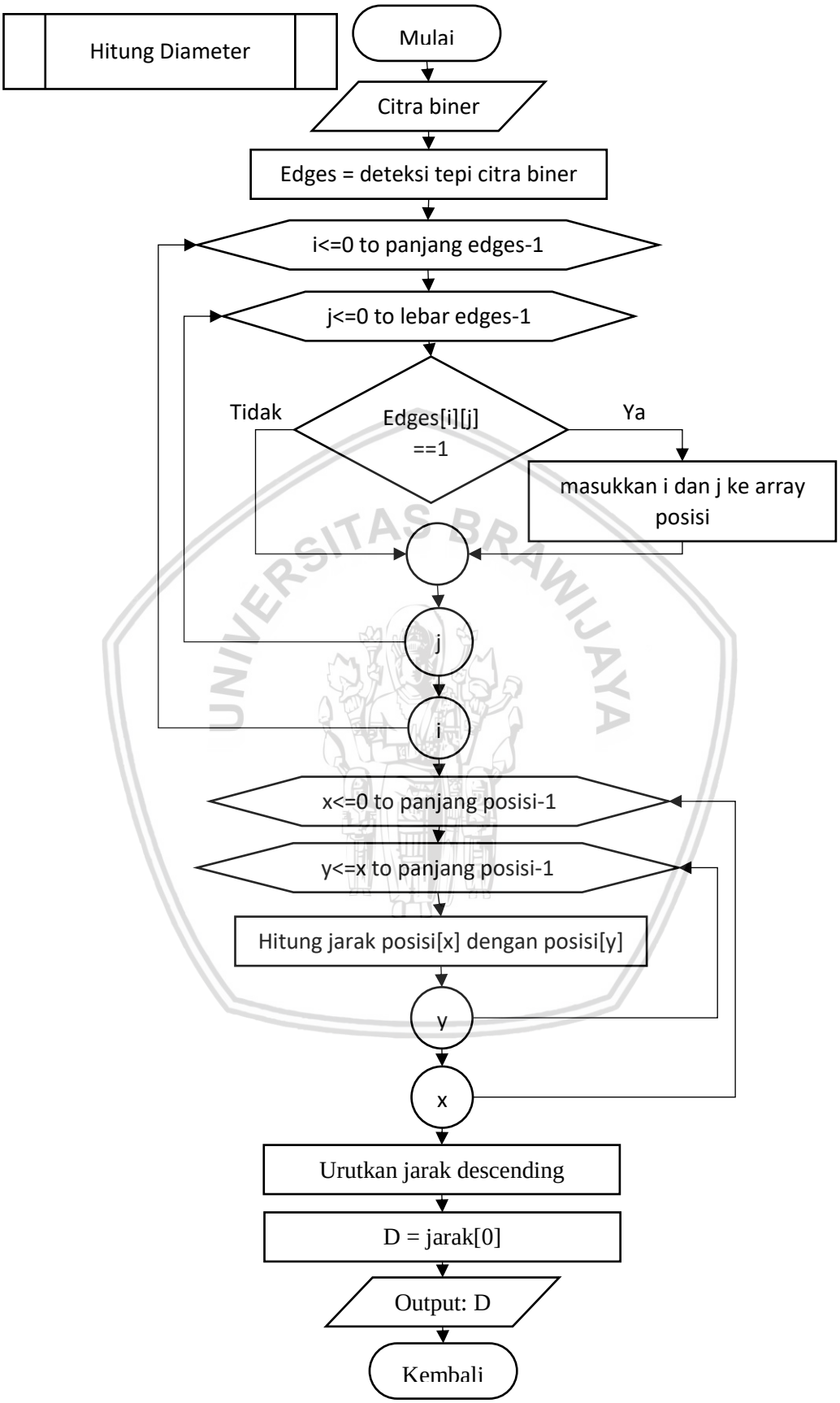
Fitur *minor axis length* didapatkan dengan mengukur lebar objek makanan. Citra masukan yang digunakan adalah citra biner. Gambar 4.11 menunjukkan proses untuk mendapatkan fitur *minor axis length*.



Gambar 4.11 Diagram Alir Hitung *Minor Axis Length*

4.1.2.3 Diameter

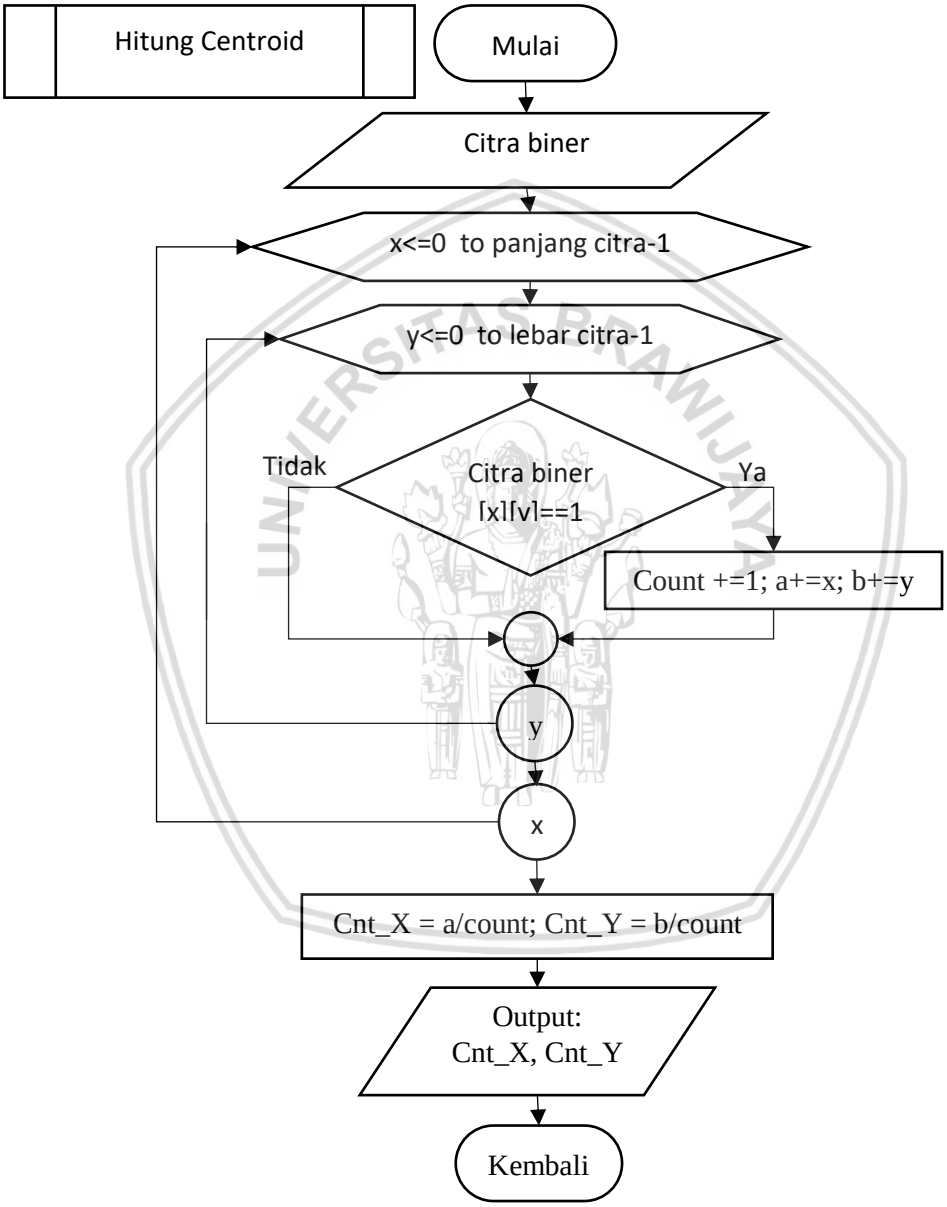
Fitur diameter menunjukkan jarak terpanjang antar tepi pada objek makanan. Untuk mendapatkan fitur diameter dilakukan deteksi tepi pada citra biner menggunakan metode *Canny Edge Detection*. Setelah dilakukan perulangan untuk menyimpan koordinat citra yang bernilai 1. Kemudian setiap koordinat dihitung jaraknya dengan koordinat yang lain. Jarak terpanjang digunakan sebagai nilai diameter. Proses mendapatkan fitur diameter dapat dilihat pada Gambar 4.12.



Gambar 4.12 Diagram Alir Hitung *Diameter*

4.1.2.4 *Centroid*

Centroid merepresentasikan titik koordinat yang terletak pada tengah objek makanan. Untuk mendapatkan nilai *centroid*, citra masukan yang digunakan adalah citra biner. Untuk mendapatkan koordinat titik tengah dilakukan perhitungan rata-rata semua koordinat citra biner yang bernilai satu. Fitur *centroid* menghasilkan dua fitur yaitu *centroid* untuk koordinat X, centroid untuk koordinat Y. Gambar 4.13 menunjukkan cara mendapatkan fitur *centroid*.

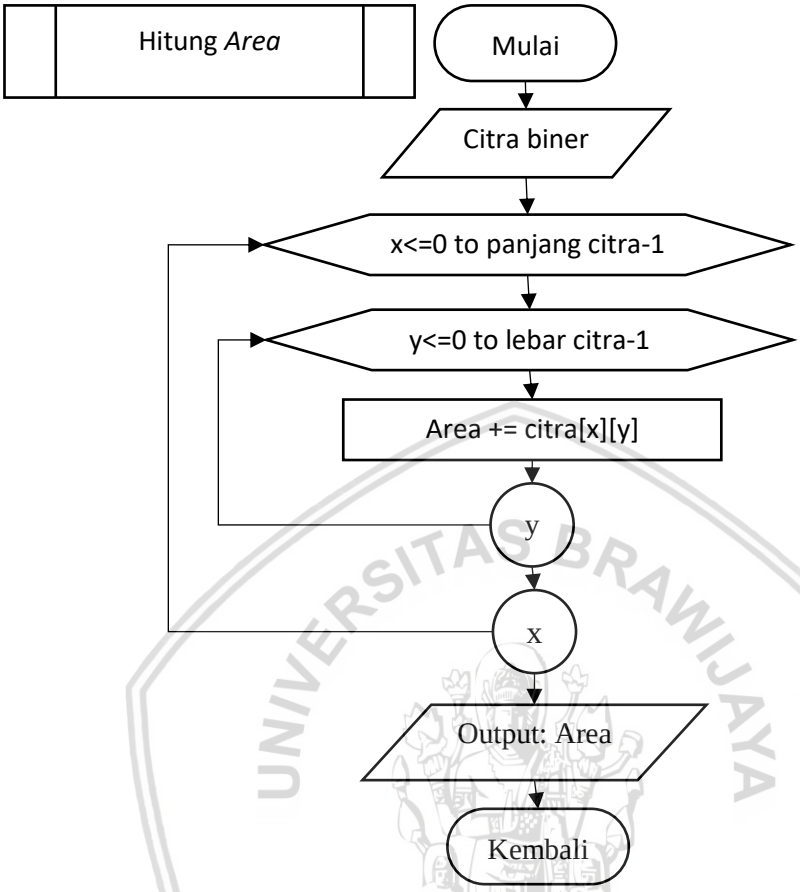


Gambar 4.13 Diagram Alir Hitung *Centroid*

4.1.2.5 *Area*

Fitur *area* menunjukkan jumlah piksel pada daerah objek. Masukan yang digunakan adalah citra biner. Untuk mendapatkan fitur *area* dilakukan perulangan

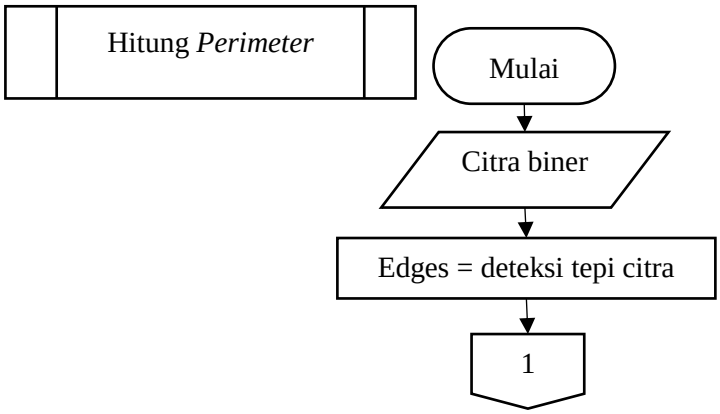
untuk menjumlahkan semua piksel pada citra biner. Gambar 4.14 menunjukkan proses mendapatkan fitur *area*.

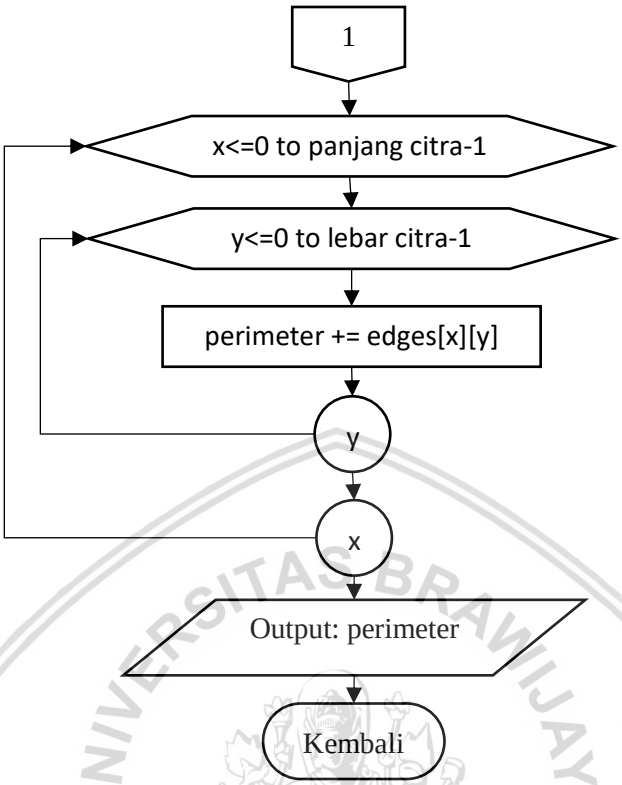


Gambar 4.14 Diagram Alir Hitung *Area*

4.1.2.6 *Perimeter*

Fitur *perimeter* menunjukkan jumlah piksel pada daerah tepi objek. Untuk mendapatkan fitur area pertama dilakukan deteksi tepi menggunakan metode *Canny Edge Detection*. Setelah itu dilakukan perulangan untuk menjumlahankan semua piksel pada citra hasil deteksi tepi. Gambar 4.15 menunjukkan proses mendapatkan fitur *perimeter*.

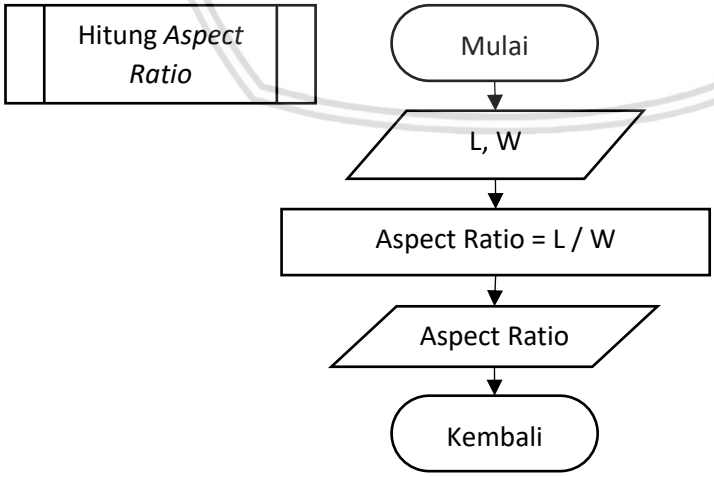




Gambar 4.15 Diagram Alir Hitung *Perimeter*

4.1.2.7 *Aspect Ratio*

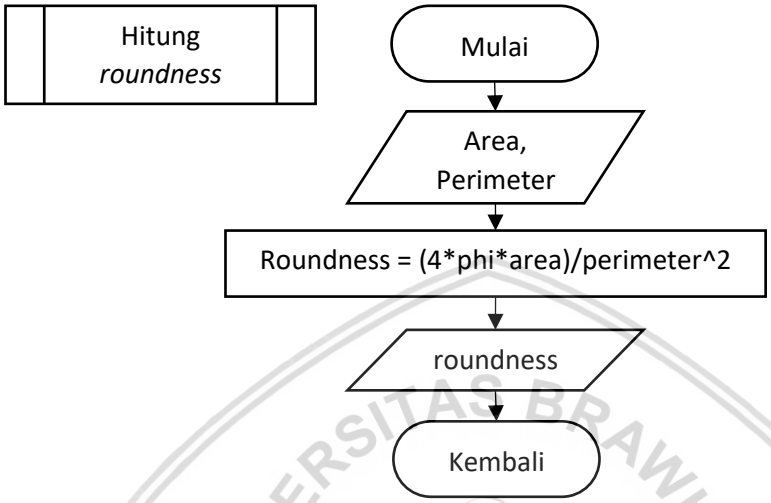
Untuk mendapat nilai fitur *aspect ratio* diperlukan masukan berupa *major axis length* dan *minor axis length*. Kemudian dilakukan perhitungan nilai *aspect ratio* dengan cara membagi nilai *major axis length* dengan *minor axis length*. Gambar 4.16 menunjukkan proses mendapatkan fitur *aspect ratio*.



Gambar 4.16 Diagram Alir Hitung *Aspect Ratio*

4.1.2.8 *Roundness*

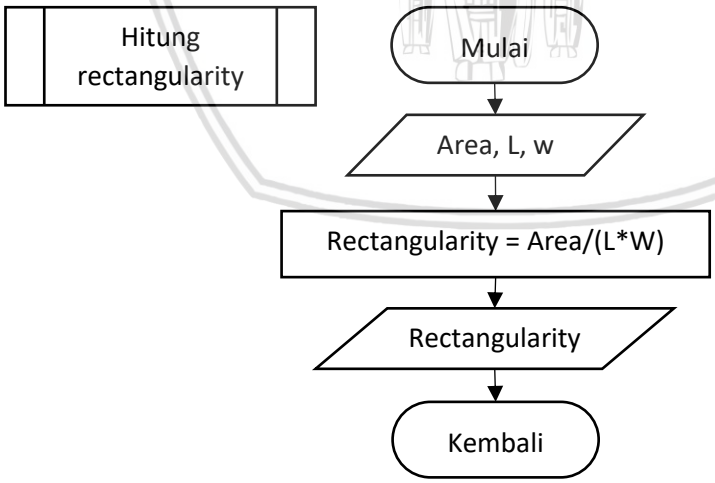
Fitur *roundness* digunakan untuk mengukur seberapa bundar bentuk objek makanan. Proses ini membutuhkan masukan berupa nilai *area* dan *perimeter*. Setelah itu dihitung menggunakan rumus pada Persamaan 2.15. Gambar 4.17 menunjukkan proses mendapatkan fitur *roundness*.



Gambar 4.17 Diagram Alir Hitung *Roundness*

4.1.2.9 *Rectangularity*

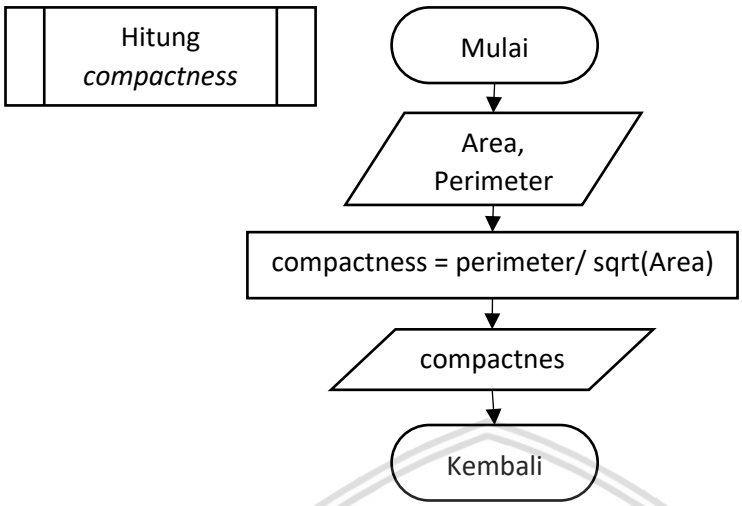
Proses ini mendapatkan fitur *rectangularity* membutuhkan masukan berupa nilai *area, major axis length* dan *minor axis length*. Setelah itu dihitung menggunakan rumus pada Persamaan 2.16. Gambar 4.18 menunjukkan proses mendapatkan fitur *rectangularity*.



Gambar 4.18 Diagram Alir Hitung *Rectangularity*

4.1.2.10 *Compactness*

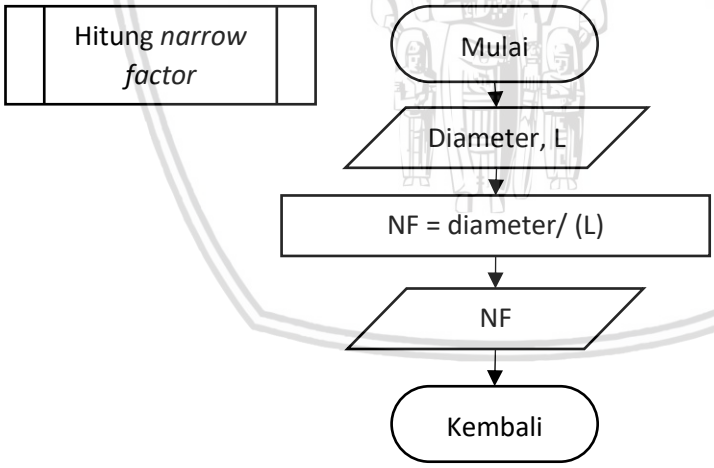
Untuk mendapatkan fitur *compactness* membutuhkan masukan berupa nilai *area* dan *perimeter*. Setelah itu dihitung menggunakan rumus pada Persamaan 2.17. Gambar 4.19 menunjukkan proses mendapatkan fitur *compactness*.



Gambar 4.19 Diagram Alir Hitung *Compactness*

4.1.2.11 *Narrow Factor*

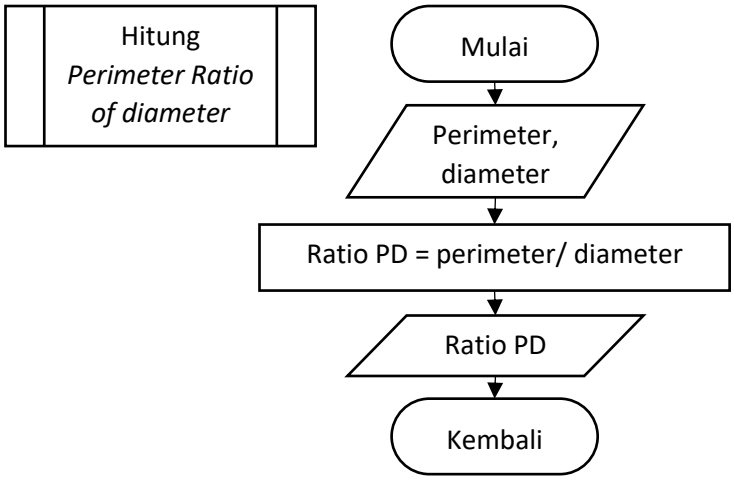
Narrow factor merupakan perbandingan antara diameter dan *major axis length* sesuai dengan Persamaan 2.18. Oleh sebab itu sebagai masukan diperlukan nilai diameter dan *major axis length*. Untuk melihat proses mendapatkan fitur *narrow factor* dapat dilihat pada Gambar 4.20.



Gambar 4.20 Diagram Alir Hitung *Narrow Factor*

4.1.2.12 Rasio *Perimeter* dengan *Diameter*

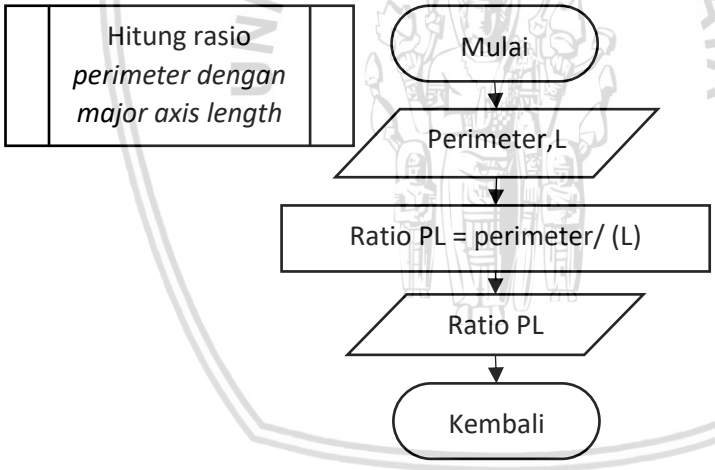
Untuk mendapatkan fitur rasio *perimeter* dengan *diameter* diperlukan masukan nilai *perimeter* dan *diameter*. Perhitungan rasio *perimeter* dengan *diameter* sesuai dengan Persamaan 2.19. Untuk melihat proses mendapatkan fitur rasio *perimeter* dengan *diameter* dapat dilihat pada Gambar 4.21.



Gambar 4.21 Diagram Alir Hitung Rasio *Perimeter* dengan *Diameter*

4.1.2.13 Rasio *Perimeter* dengan *Major Axis Length*

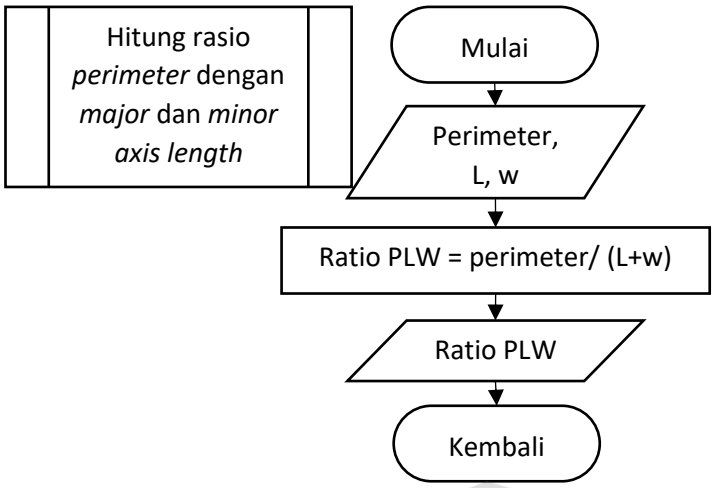
Untuk mendapatkan fitur rasio *perimeter* dengan *major axis length* diperlukan masukan nilai *diameter* dan *major axis length*. Untuk melihat proses mendapatkan fitur rasio *perimeter* dengan *major axis length* dapat dilihat pada Gambar 4.22.



Gambar 4.22 Diagram Alir Hitung Rasio *Perimeter* dengan *Major Axis Length*

4.1.2.14 Rasio *Perimeter* dengan *Major Axis Length* dan *Minor Axis Length*

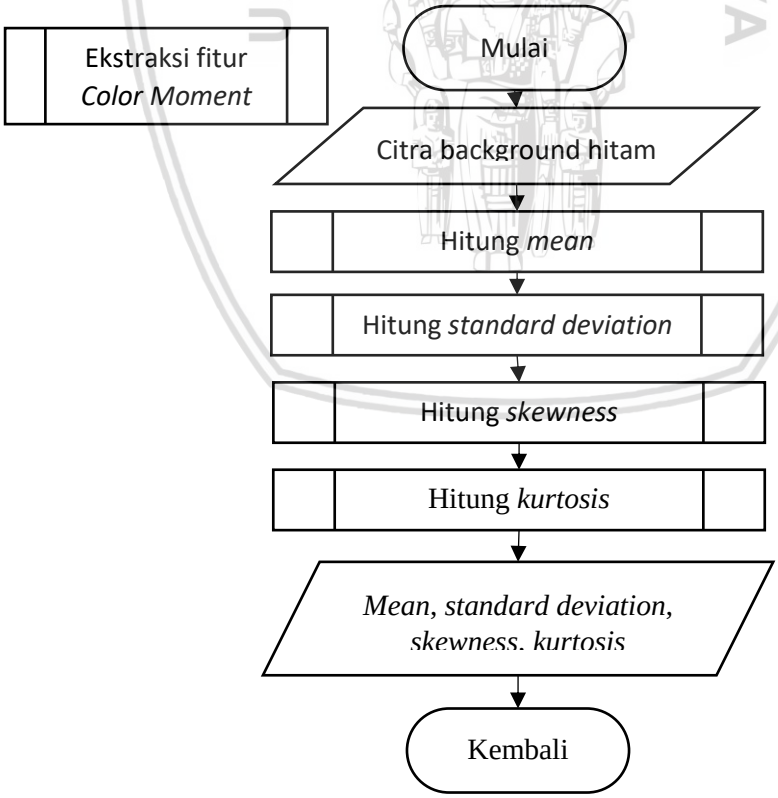
Untuk mendapatkan fitur rasio *perimeter* dengan *major axis length* dan *minor axis length* diperlukan masukan nilai *diameter*, *major axis length* dan *minor axis length*. Untuk melihat proses mendapatkan fitur rasio *perimeter* dengan *major axis length* dan *minor axis length* dapat dilihat pada Gambar 4.23.



Gambar 4.23 Diagram Alir Hitung Rasio *Perimeter dengan Major Axis Length* dan *Minor Axis Length*

4.1.3 Ekstraksi Fitur *Color Moment*

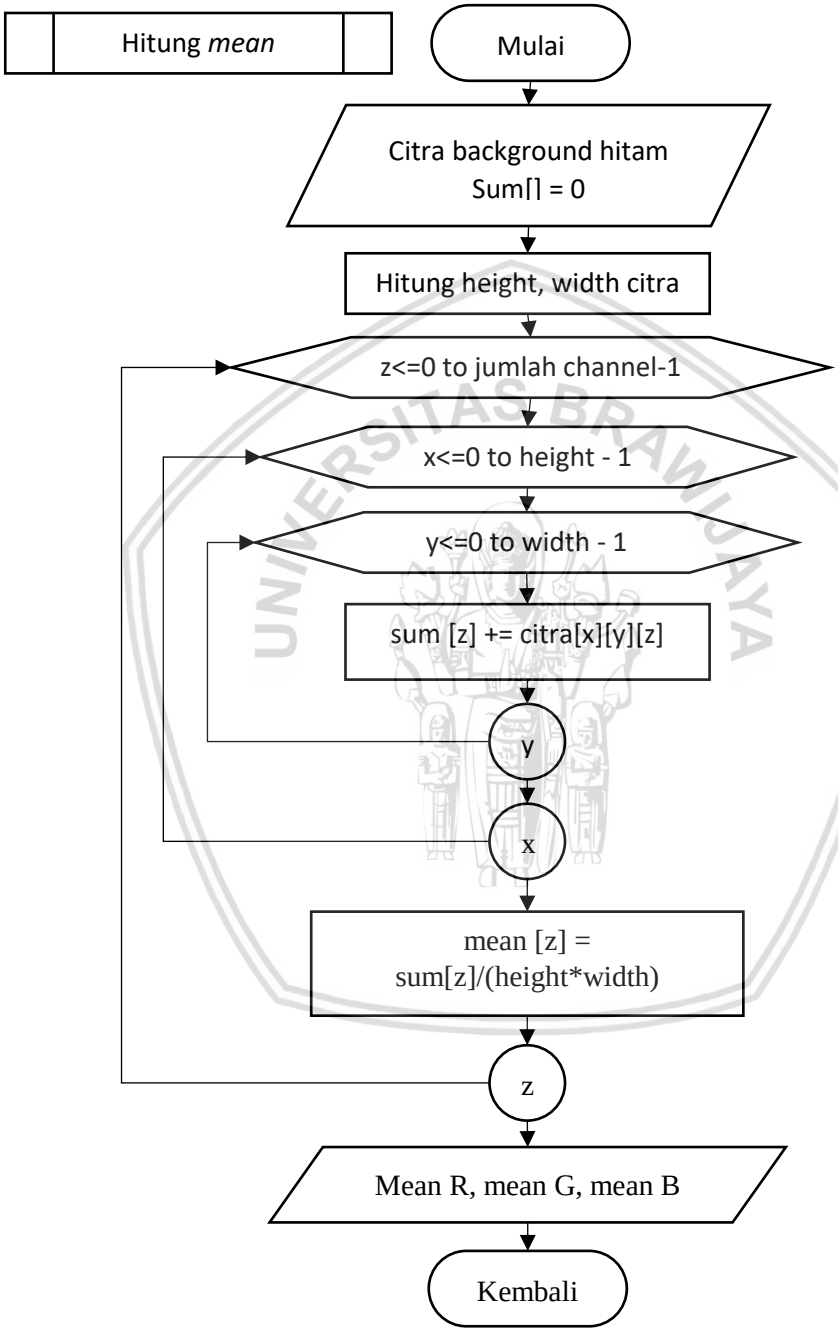
Ekstraksi fitur *Color Moment* didapatkan dengan menghitung nilai distribusi warna RGB terhadap citra tersegmentasi yang dihasilkan pada proses sebelumnya. Distribusi warna yang dihitung meliputi *mean*, *standard deviation*, *skewness* dan *kurtosis*. Proses ekstraksi fitur *Color Moment* dapat dilihat pada Gambar 4.24.



Gambar 4.24 Diagram Alir Ekstraksi Fitur *Color Moment*

4.1.3.1 *Mean*

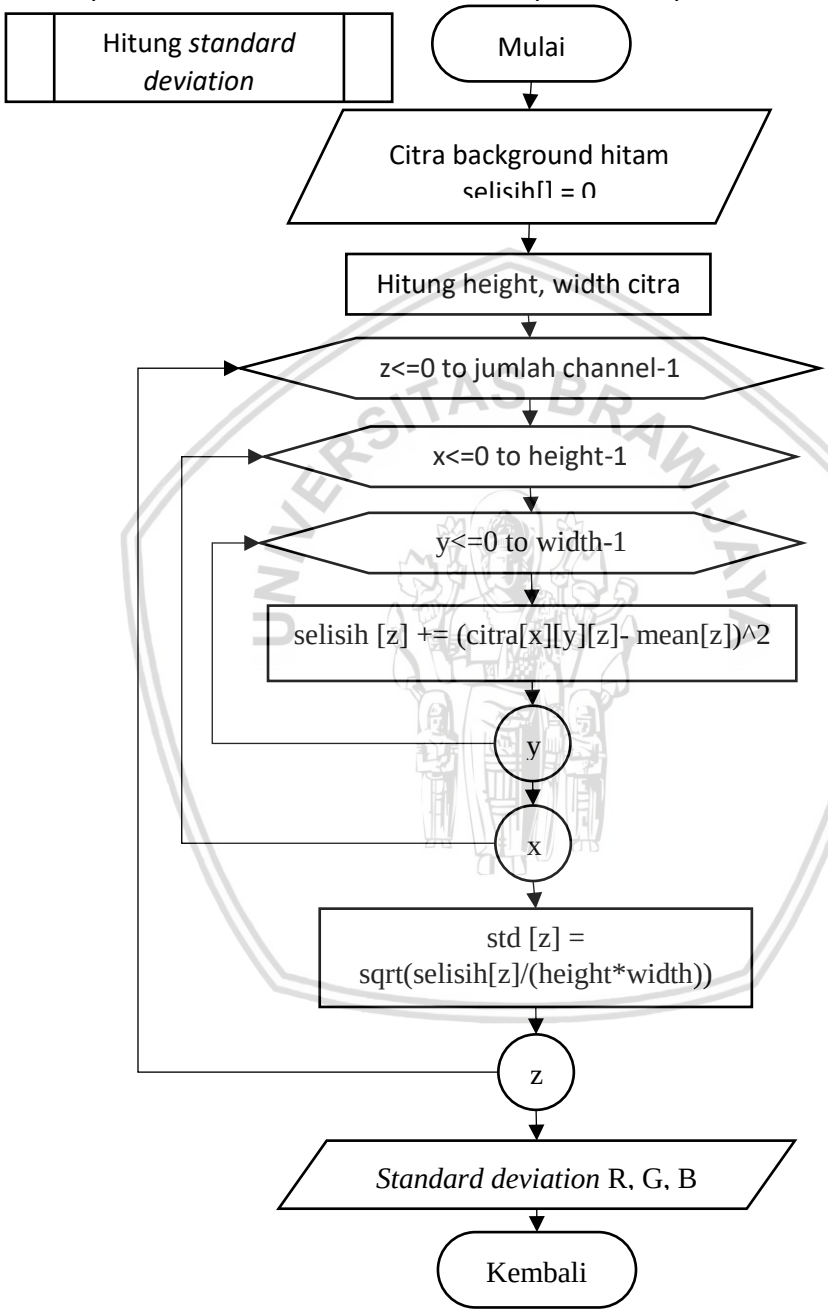
Fitur *mean* didapatkan dengan cara menjumlahkan setiap nilai piksel citra RGB yang telah disegmentasi. Kemudian membagi hasil penjumlahan tersebut dengan perkalian panjang dan lebar citra. Proses tersebut dilakukan untuk setiap *Channel* citra RGB. Gambar 4.25 menunjukkan diagram alir ekstraksi fitur *mean*.



Gambar 4.25 Diagram Alir Hitung *Mean*

4.1.3.2 *Standard deviation*

Untuk mendapatkan fitur *standard deviation* setiap *channel* warna RGB dilakukan perulangan untuk menjumlahkan kuadrat selisih piksel citra dengan nilai *mean* yang didapatkan pada proses sebelumnya. Setelah itu nilai penjumlahan tersebut di akar kuadrat dan digunakan sebagai nilai *standard deviation*. Proses mendapatkan fitur *standard deviation* dapat dilihat pada Gambar 4.26.

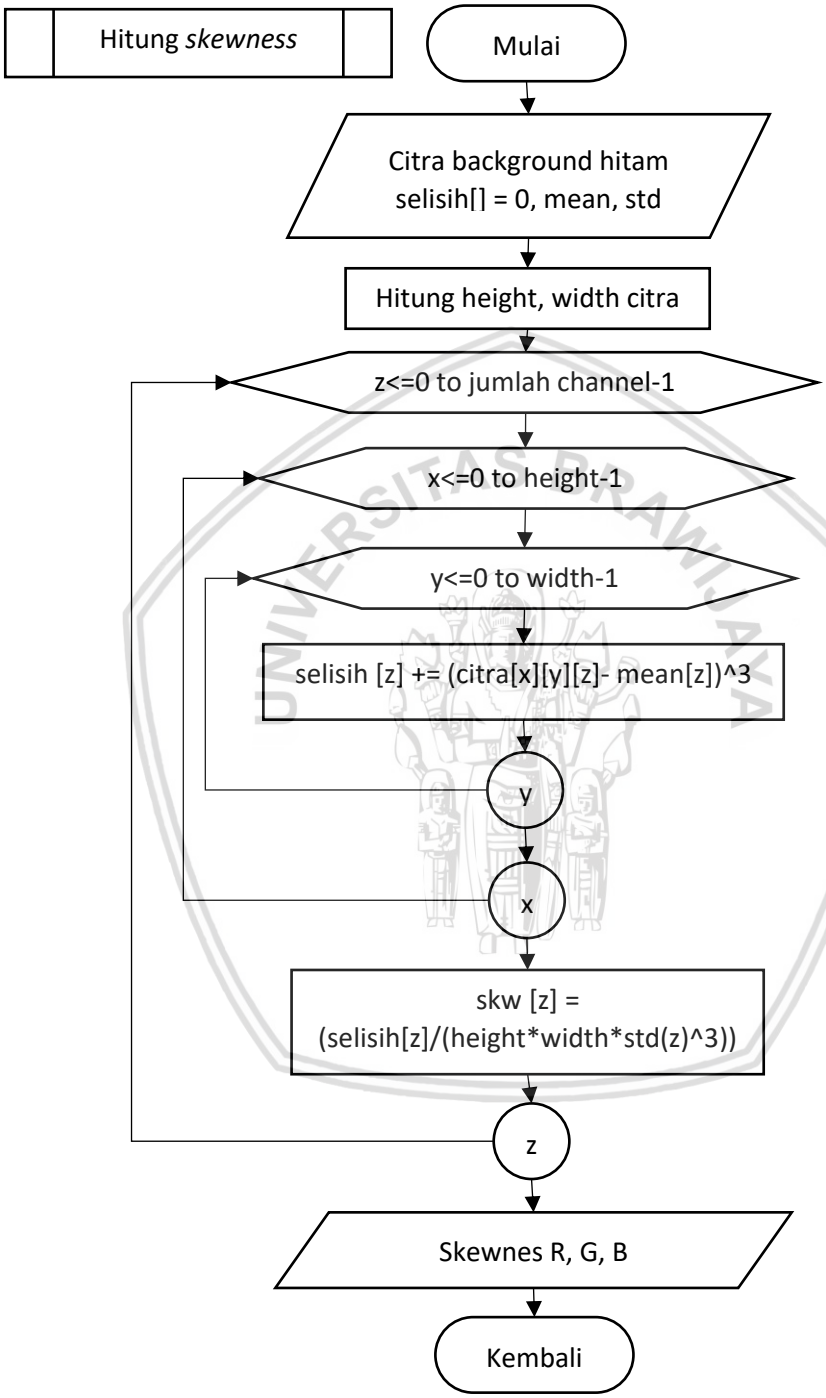


Gambar 4.26 Diagram Alir Hitung *Standard deviation*

4.1.3.3 *Skewness*

Proses mendapatkan fitur *skewness* dapat dilihat pada Gambar 4.27. Pertama dilakukan pengukuran panjang dan lebar citra tersegmentasi. Kemudian

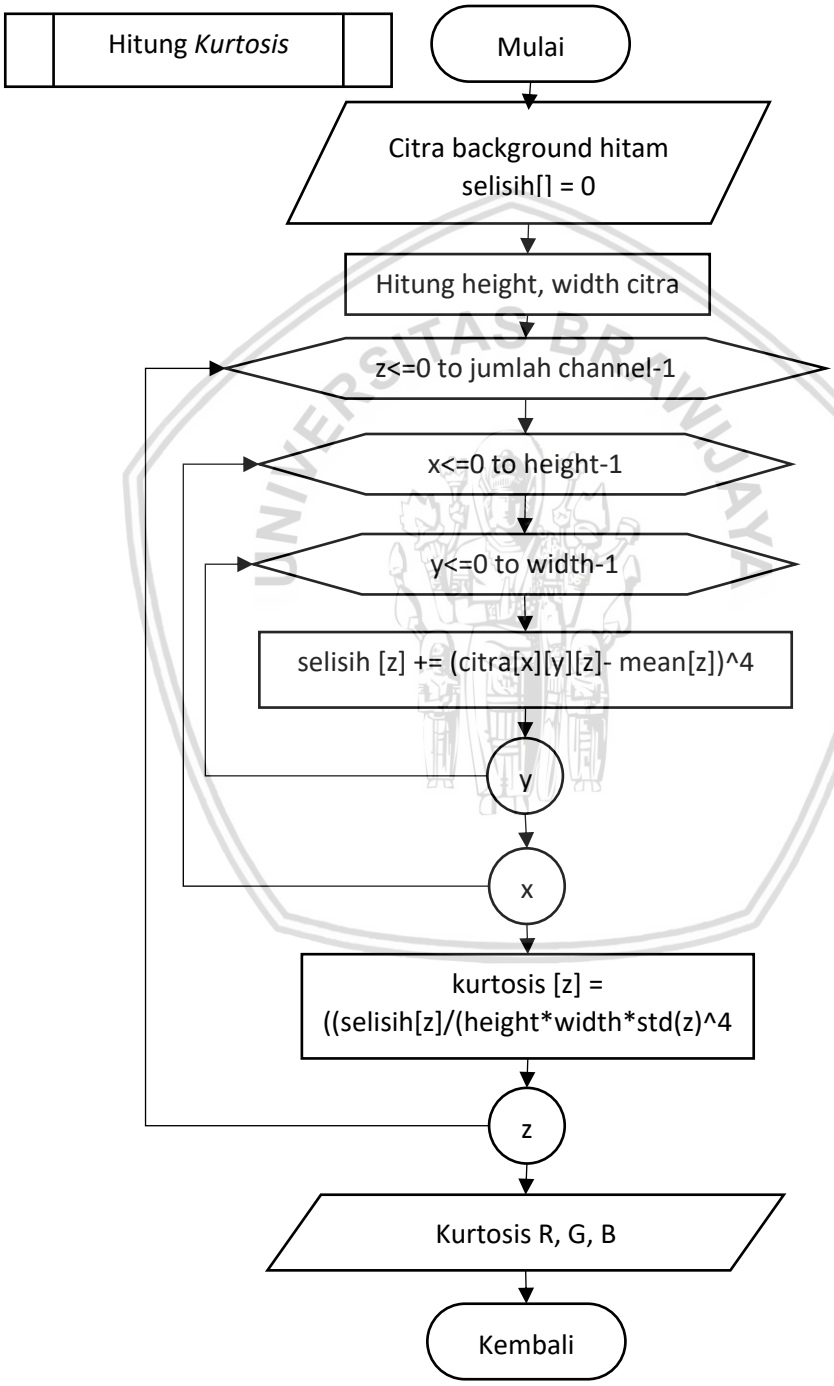
dilakukan perulangan sejumlah channel pada warna RGB. Setelah itu dilakukan perulangan untuk menjumlahkan kubik selisih piksel citra dengan nilai *mean* yang didapatkan pada proses sebelumnya. Setelah itu nilai penjumlahan tersebut di akar pangkat tiga dan digunakan sebagai nilai *skewness*.



Gambar 4.27 Diagram Alir Hitung *Skewness*

4.1.3.4 *Kurtosis*

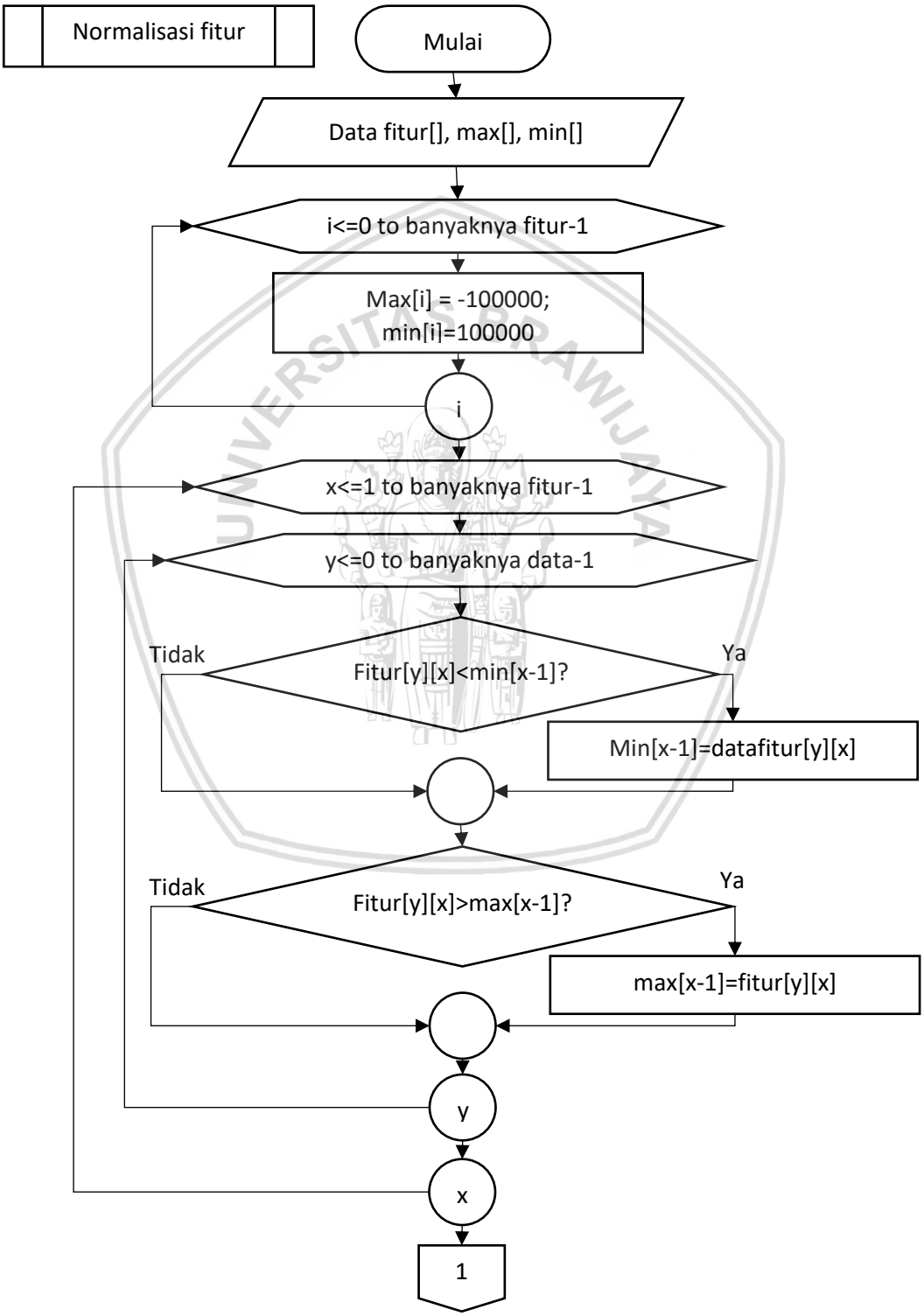
Proses mendapatkan fitur *kurtosis* dapat dilihat pada Gambar 4.28. Pertama dilakukan pengukuran panjang dan lebar citra tersegmentasi. Kemudian dilakukan perulangan sejumlah *channel* pada warna RGB. Setelah itu dilakukan perulangan untuk menjumlahkan pangkat empat selisih piksel citra dengan nilai *mean* yang didapatkan pada proses sebelumnya. Setelah itu nilai penjumlahan tersebut di akar pangkat empat dan digunakan sebagai nilai *kurtosis*.

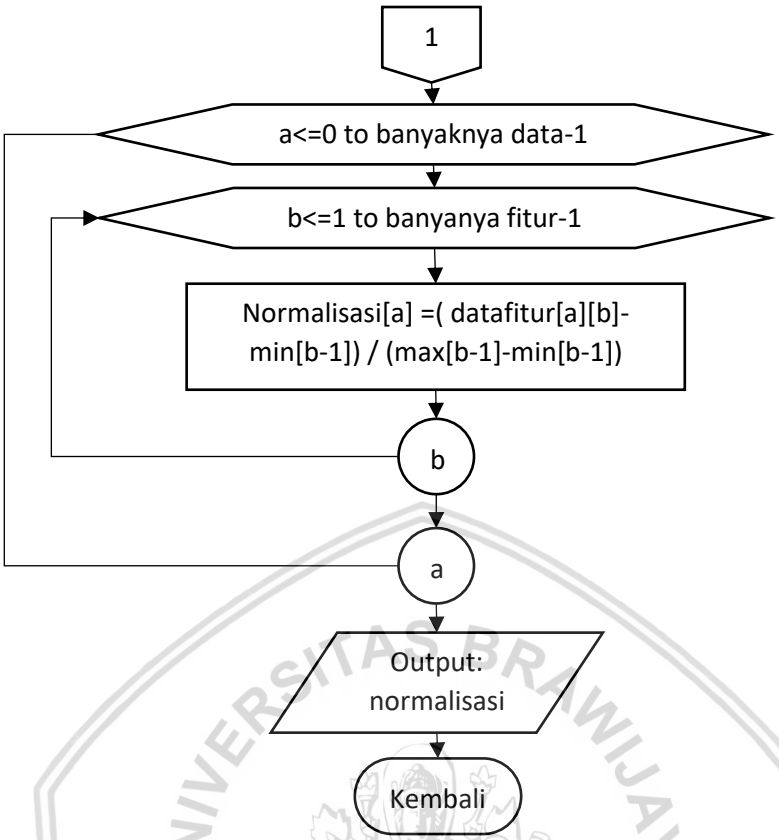


Gambar 4.28 Diagram Alir Hitung *Kurtosis*

4.1.4 Normalisasi Fitur

Normalisasi fitur digunakan untuk membuat data memiliki rentang nilai yang sama. Metode yang digunakan adalah *min-max normalization*. Metode ini menggunakan nilai minimal dan nilai maksimal setiap fitur untuk melakukan normalisasi. Proses perhitungan nilai normalisasi dapat dilihat pada Persamaan 2.27. Proses normalisasi fitur dapat dilihat pada Gambar 4.29.

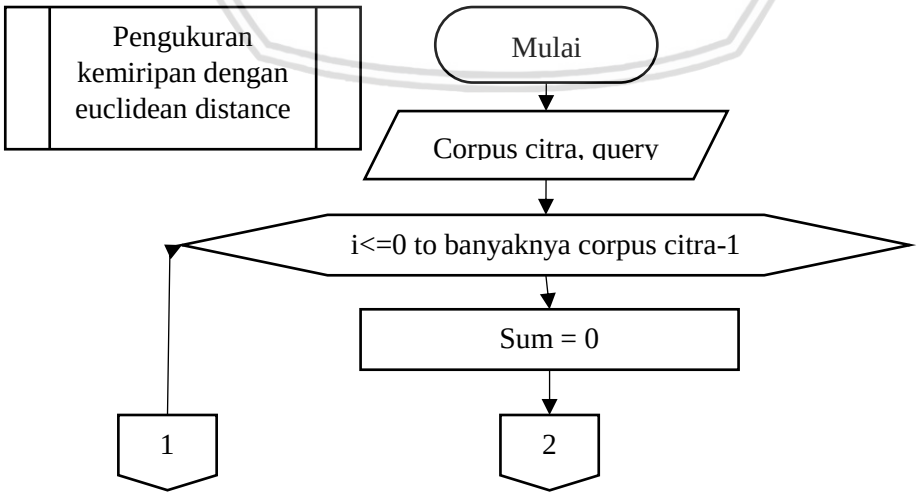


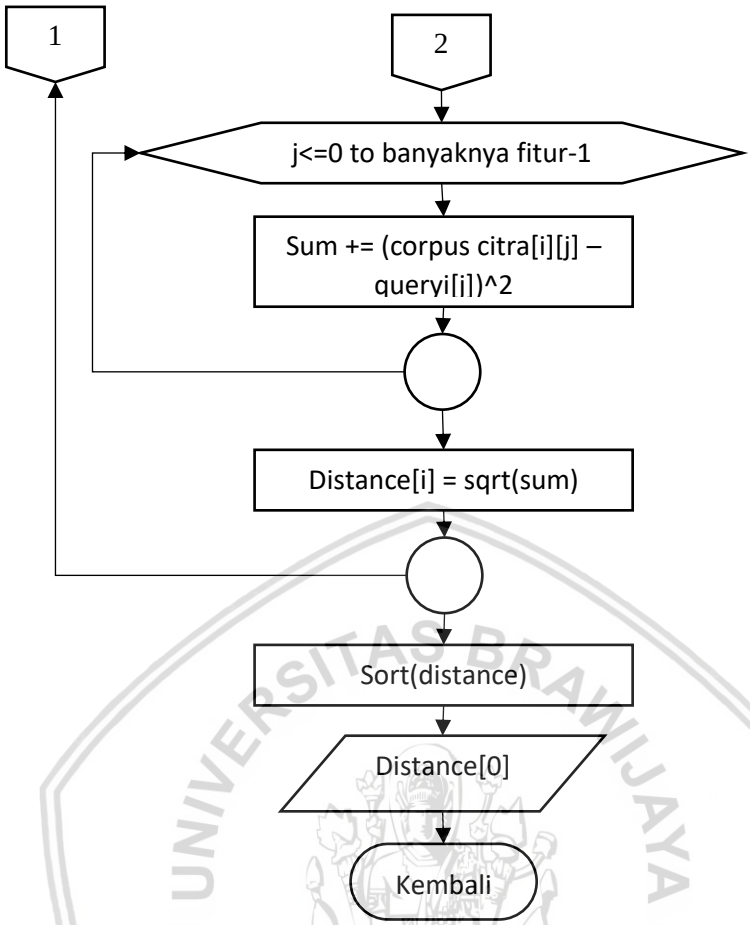


Gambar 4.29 Diagram Alir Normalisasi Fitur

4.1.5 Pengukuran Kemiripan dengan *Euclidean Distance*

Setelah dilakukan normalisasi fitur, tahapan selanjutnya yaitu pengukuran kemiripan antara citra *query* dengan setiap data *corpus* citra. Proses perhitungan kemiripan menggunakan metode *Euclidean Distance*. Kemudian hasil pengukuran tersebut diurutkan dari terkecil ke terbesar. Gambar 4.30 menunjukkan proses pengukuran kemiripan dengan *Euclidean Distance*.





Gambar 4.30 Diagram Alir Pengukuran Kemiripan

4.2 Perhitungan Manual

Tahap ini memberikan gambaran umum proses pencarian tanpa menggunakan bantuan sistem. Langkah-langkah perhitungan data citra dilakukan secara detail dengan menggunakan MS. Excel. Pada perhitungan manual ini yang digunakan adalah gambar donat yang berukuran 10x10 piksel. Gambar 4.31 menunjukkan citra donat yang digunakan pada perhitungan manual ini.

Gambar 4.31 Citra makanan perhitungan manual

4.2.1 Pre-processing

Tahap pertama yaitu *pre-processing* yang terdiri dari proses *resize* citra, *filtering*, binerisasi, *cropping* dan segmentasi citra. Tahap ini menghasilkan citra biner digunakan untuk proses ekstraksi fitur bentuk dan citra tersegmentasi digunakan untuk ekstraksi fitur warna.

4.2.1.1 Perhitungan *Resize* Citra

Proses *resize* pada perhitungan manual ini dilakukan dengan mengubah citra awal menjadi berukuran 10 x 10 piksel. Gambar 4.32 menunjukkan hasil gambar proses resize citra.

(a) Sebelum Proses Resize Citra (a) Setelah Proses Resize Citra

Gambar 4.32 Perubahan Citra Makanan Proses *Resize* Citra

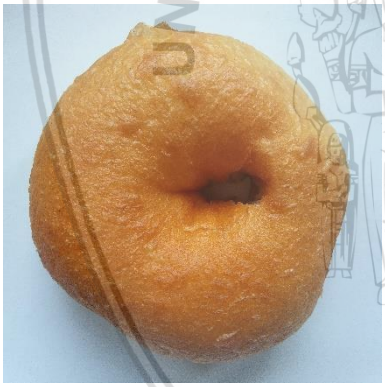
4.2.1.2 Perhitungan *Filtering*

Proses *filtering* pada penelitian ini menggunakan metode *Gaussian filter* dengan mask berupa matriks berukuran 3x3. *Gaussian filter* bekerja menggunakan fungsi Gaussian pada Persamaan 2.1. *Gaussian Filter* Hasil proses *filtering* ditunjukkan pada Tabel 4.1. Gambar 4.33 menunjukkan hasil proses *filtering*.

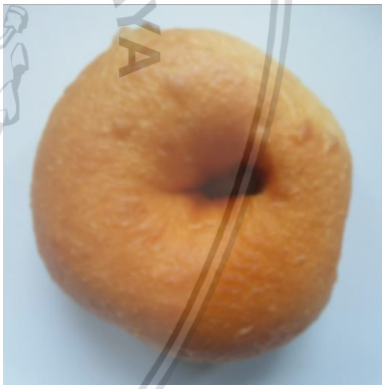
Tabel 4.1 Hasil Perhitungan *Gaussian Filtering*

R/ G/ B	1	2	3	4	5	6	7	8	9	10
1	197/ 217/ 228	198/ 210/ 212	200/ 195/ 176	207/ 192/ 162	215/ 201/ 173	216/ 206/ 181	214/ 213/ 197	210/ 224/ 224	204/ 225/ 236	200/ 222/ 234
2	200/ 217/ 223	200/ 205/ 197	199/ 179/ 144	202/ 168/ 117	208/ 174/ 125	210/ 180/ 135	210/ 192/ 157	210/ 212/ 199	206/ 224/ 229	202/ 224/ 235
3	206/ 210/ 201	203/ 190/ 163	196/ 154/ 99	190/ 133/ 65	187/ 125/ 58	187/ 126/ 65	195/ 146/ 89	208/ 183/ 141	209/ 212/ 198	206/ 221/ 222

4	206/193/170	202/169/129	190/133/72	175/110/46	159/87/28	147/76/26	160/98/44	191/145/86	204/186/149	202/202/186
5	195/173/146	191/148/102	183/116/48	171/97/28	152/74/15	135/61/16	142/79/30	172/115/54	189/156/112	190/179/156
6	181/158/136	178/131/90	179/107/40	183/106/33	177/97/25	169/93/29	170/102/39	175/112/48	179/140/99	181/166/147
7	174/162/154	168/133/106	171/107/52	188/115/46	192/113/38	187/108/33	183/110/38	178/116/56	175/147/118	176/172/164
8	172/184/191	160/155/150	153/116/84	164/102/49	170/96/32	166/90/24	161/91/30	161/116/75	169/164/157	174/190/200
9	175/200/217	161/180/192	140/136/132	132/101/78	133/86/50	131/82/42	131/90/57	142/126/113	162/176/186	172/199/219
10	178/205/223	165/190/207	138/150/160	120/111/107	114/91/75	111/85/65	116/98/85	134/136/138	159/180/195	171/199/219



(b)Sebelum Proses Filtering



(c) Setelah Proses Filtering

Gambar 4.33 Perubahan Citra Makanan Proses *Filtering*

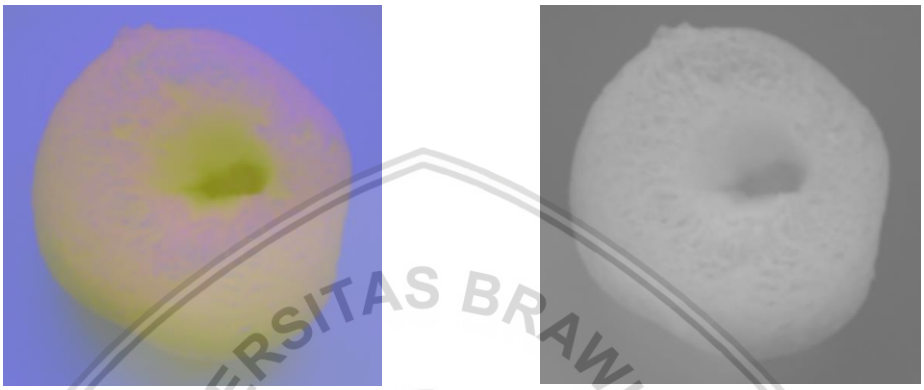
4.2.1.3 Konversi RGB ke L*a*b

Tahap ini dimulai dengan mengubah citra RGB blur menjadi citra dengan color space L*a*b. Kemudian diambil komponen B dan ditempatkan pada satu matriks tersendiri. Matriks komponen B dapat dilihat pada Tabel 4.2. Gambar 4.34 menunjukkan hasil konversi RGB ke L*a*b.

Tabel 4.2 Hasil perhitungan komponen B (L*a*b)

120	126	138	145	144	142	136	126	120	120
123	131	149	159	158	155	148	134	124	120
132	143	163	172	173	170	165	153	135	126

141	153	169	173	173	169	168	166	149	136
145	158	175	177	174	168	167	170	156	141
143	157	176	179	179	175	173	173	156	140
134	148	168	176	180	180	178	170	147	133
123	131	151	167	175	176	173	157	132	121
117	120	131	146	157	160	154	137	121	115
116	117	121	131	141	144	138	127	118	115



(a) Citra Makanan L*a*b (b) komponen b* pada Citra L*a*b

Gambar 4.34 Hasil Konversi RGB ke L*a*b

4.2.1.4 Perhitungan Binerisasi

Proses selanjutnya yaitu perhitungan binerisasi yang dimulai dengan melakukan proses *thresholding* komponen B citra L*a*b dengan metode Otsu. Hasil proses *thresholding* dapat dilihat pada Tabel 4.3. Untuk menyempurnakan hasil *thresholding* tersebut tahap selanjutnya dilakukan operasi morfologi *closing*. Operasi morfologi *closing* menggunakan kernel yang berukuran 3x3. Hasilnya dapat dilihat pada Tabel 4.4. Tahap terakhir yaitu mengubah piksel yang bernilai 255 menjadi 1 agar menjadi bentuk biner yang dapat dilihat pada Tabel 4.5. Gambar 4.35 menunjukkan hasil binerisasi.

Tabel 4.3 Hasil *thresholding* dengan metode Otsu

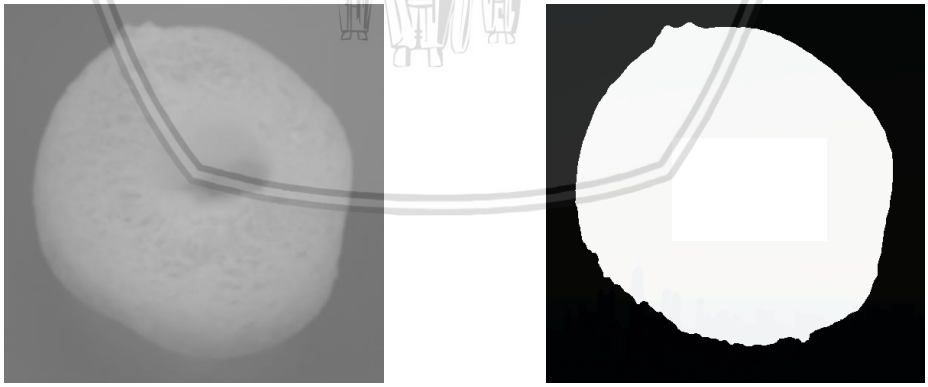
0	0	0	0	0	0	0	0	0	0
0	0	0	255	255	255	0	0	0	0
0	0	255	255	255	255	255	255	0	0
0	255	255	255	255	255	255	255	0	0
0	255	255	255	255	255	255	255	255	0
0	255	255	255	255	255	255	255	255	0
0	0	255	255	255	255	255	255	0	0
0	0	255	255	255	255	255	255	0	0
0	0	0	0	255	255	255	0	0	0
0	0	0	0	0	0	0	0	0	0

Tabel 4.4 Hasil perhitungan operasi morfologi *closing*

0	0	0	0	0	0	0	0	0	0
0	0	0	255	255	255	0	0	0	0
0	0	255	255	255	255	255	255	0	0
0	255	255	255	255	255	255	255	0	0
0	255	255	255	255	255	255	255	255	0
0	255	255	255	255	255	255	255	255	0
0	0	255	255	255	255	255	255	0	0
0	0	255	255	255	255	255	255	0	0
0	0	0	0	255	255	255	0	0	0
0	0	0	0	0	0	0	0	0	0

Tabel 4.5 Hasil Perhitungan Binerisasi

0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	0	0
0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0



(a) Sebelum dilakukan binerisasi (b) Hasil Proses Binerisasi

Gambar 4.35 Perubahan Citra Makanan Proses Binerisasi

4.2.1.5 Perhitungan *Cropping*

Proses cropping dilakukan agar citra berisi objek makanan saja. Proses cropping dilakukan untuk citra biner dan citra makanan RGB. Proses *cropping* mengubah citra berukuran 10x10 piksel menadi 8x8 piksel. Masing-masing hasil

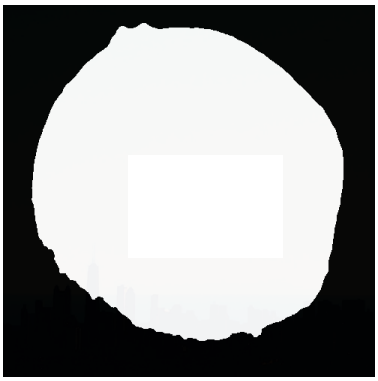
cropping citra biner dan citra RGB dapat dilihat pada Tabel 4.6 dan Tabel 4.7. Gambar 4.36 menunjukkan hasil *cropping*.

Tabel 4.6 Hasil Perhitungan *Cropping*

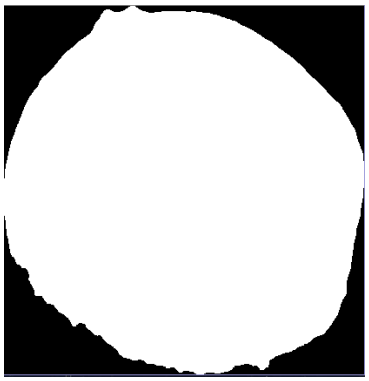
0	0	1	1	1	0	0	0
0	1	1	1	1	1	1	0
1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	0
0	0	0	1	1	1	0	0

Tabel 4.7 Hasil Perhitungan *Cropping*

R/G/B	1	2	3	4	5	6	7	8
1	202/222/231	189/148/87	201/148/72	224/180/118	217/178/121	214/185/129	214/233/240	206/229/238
2	210/206/175	199/151/88	191/134/57	180/111/36	192/129/60	194/135/75	215/179/111	209/229/238
3	220/161/99	173/116/62	185/121/64	155/80/22	143/67/18	156/81/26	218/171/93	215/184/120
4	196/132/61	188/121/50	166/89/9	130/48/1	101/26/4	68/21/7	181/117/46	180/128/71
5	160/89/32	184/113/46	180/94/16	194/114/25	180/8919	211/154/77	169/95/24	177/122/66
6	157/89/49	170/96/31	214/146/81	204/128/54	187/110/32	193/122/50	187/115/41	173/122/78
7	145/165/176	145/91/47	178/105/37	172/85/16	178/94/16	155/72/6	156/94/34	167/190/208
8	165/191/209	119/129/139	113/71/46	132/71/27	133/70/21	129/75/31	135/106/92	166/195/213



(a) Sebelum dilakukan *cropping*



(b) Citra Biner Setelah dilakukan cropping

(c) Citra RGB setelah dilakukan cropping

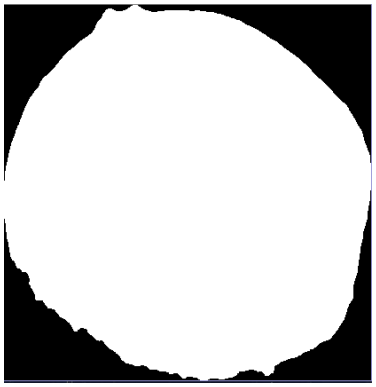
Gambar 4.36 Perubahan Citra Makanan Proses *Cropping*

4.2.1.6 Perhitungan Segmentasi Citra

Segmentasi citra dilakukan dengan cara mengalikan setiap piksel citra biner pada Tabel 4.6 dengan setiap piksel citra RGB pada Tabel 4.7. Hasilnya perkalian tersebut menghasilkan citra tersegmentasi yang memiliki *background* berwarna hitam. Tabel 4.8 menunjukkan hasil perhitungan segmentasi. Gambar 4.37 menunjukkan hasil segmentasi citra.

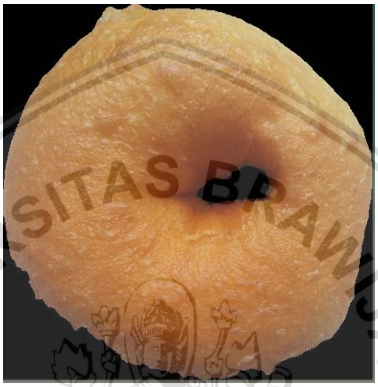
Tabel 4.8 Hasil Perhitungan Segmentasi Citra

R/G/B	1	2	3	4	5	6	7	8
1	0/0/0	0/0/0	201/148/72	224/180/118	217/178/121	0/0/0	0/0/0	0/0/0
2	0/0/0	199/151/88	191/134/57	180/111/36	192/129/60	194/135/75	215/179/111	0/0/0
3	220/161/99	173/116/62	185/121/64	155/80/22	143/67/18	156/81/26	218/171/93	0/0/0
4	196/132/61	188/121/50	166/89/9	130/48/1	101/26/4	68/21/7	181/117/46	180/128/71
5	160/89/32	184/113/46	180/94/16	194/114/25	180/89/19	211/154/77	169/95/24	177/122/66
6	0/0/0	170/96/31	214/146/81	204/128/54	187/110/32	193/122/50	187/115/41	0/0/0
7	0/0/0	145/91/47	178/105/37	172/85/16	178/94/16	155/72/6	156/94/34	0/0/0
8	0/0/0	0/0/0	0/0/0	132/71/27	133/70/21	129/75/31	0/0/0	0/0/0



(a) Citra biner

(b) Citra Makanan RGB



(c) Citra Tersegmentasi

Gambar 4.37 Perubahan Proses Segmentasi Citra

4.2.2 Perhitungan Ekstraksi Bentuk *Simple Morphological Shape Descriptors*

Citra biner yang dihasilkan pada *pre-processing* digunakan untuk ekstraksi fitur bentuk. Fitur bentuk yang diekstraksi meliputi *major axis length, minor axis length, diameter, centroid X, centroid Y, area, perimeter, aspect ratio, roundness, rectangularity, compactness, narrow factor,* rasio *perimeter* dengan *diameter,* rasio *perimeter* dengan *major axis length,* rasio *perimeter* dengan *major* dan *minor axis length.*

4.2.2.1 *Mayor Axis Length*

Perhitungan *major axis length* dilakukan dengan cara mengukur panjang dasar dan ujung objek. Pengukuran dilakukan dengan menghitung jumlah kotak secara vertikal pada citra biner yang dihasilkan pada *pre-processing*. Berdasarkan Tabel 4.6 maka nilai *major axis length* adalah 8.

4.2.2.2 *Minor Axis Length*

Perhitungan *minor axis length* dilakukan dengan mengukur lebar objek. Pengukuran dilakukan dengan menghitung jumlah kotak secara horizontal pada

citra biner yang dihasilkan pada *pre-processing*. Berdasarkan Tabel 4.6 maka nilai *minor axis length* adalah 8.

4.2.2.3 *Diameter*

Diameter merupakan jarak terpanjang antara dua titik pada tipe objek. Sebelum dilakukan perhitungan jarak, dilakukan pengambilan nilai piksel-piksel yang berada di bagian tepi objek dan menghapus piksel yang berada ditengah objek. Hasilnya dapat dilihat pada Tabel 4.9. Kemudian dilakukan perhitungan jarak koordinat setiap titik pada tepi objek dengan titik lainnya. Perhitungan jarak dilakukan dengan menggunakan metode *Euclidean distance* sesuai dengan Persamaan 2.27. Kemudian dipilih jarak terbesar sebagai nilai diameter.

Tabel 4.9 Matriks Yang Berisi Tepi Objek

x\y	1	2	3	4	5	6	7	8
1	0	0	**1**	1	1	0	0	0
2	0	1	0	0	0	1	1	0
3	1	0	0	0	0	0	1	0
4	1	0	0	0	0	0	0	1
5	1	0	0	0	0	0	0	1
6	0	1	0	0	0	0	1	0
7	0	1	1	0	0	0	1	0
8	0	0	0	1	1	**1**	0	0

Berikut contoh perhitungan jarak antara koordinat x1 (1,3) dengan x2 (8,6):

$$di = \sqrt{(8-1)^2 + (6-3)^2} = \sqrt{58} = 7{,}62$$

4.2.2.4 *Centroid*

Perhitungan centroid dimulai dengan menjumlahkan nilai koordinat piksel yang bernilai 1 pada citra biner. Setelah itu dihitung rata-ratanya sebagai koordinat titik tengah. Berikut perhitungan *centroid* untuk citra biner pada Tabel 4.6.

$$Titik\ tengah\ (x) = \frac{\begin{array}{c}3+4+5+2+3+4+5+6+7+1+2+3+4+5+6+\\ 7+1+2+3+4+5+6+7+8+1+2+3+4+5+6+7\\ +8+2+3+4+5+6+7+2+3+4+5+6+7+4+\\ 5+6\end{array}}{49}$$

$$= 4{,}2 \approx 4$$

$$Titik\ tengah\ (y) = \frac{\begin{array}{c}1+1+1+2+2+2+2+2+2+3+3+3+3+3+3+\\ 3+4+4+4+4+4+4+4+4+5+5+5+5+5+5+5\\ +5+6+6+6+6+6+6+7+7+7+7+7+7+8+\\ 8+8\end{array}}{49}$$

$$= 4{,}28 \approx 4$$

Jadi titik tengah objek makanan tersebut adalah pada koordinat (4,4).

4.2.2.5 *Area*

Area merupakan jumlah piksel pada daerah objek makanan. untuk mendapatkan menghitung nilai *area* dilakukan penjumlahan semua piksel pada citra biner. Berikut contoh menghitung *area* sesuai citra biner pada Tabel 4.10.

Tabel 4.10 Perhitungan *Area*

x\y	1	2	3	4	5	6	7	8
1	0	0	1	**1**	1	0	0	0
2	0	1	1	1	1	1	1	0
3	1	1	1	1	1	1	1	0
4	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1
6	0	1	1	1	1	1	1	0
7	0	1	1	1	1	1	1	0
8	0	0	0	1	1	1	0	0

$$A = 0+0+1+1+1+0+0+0+0+1+1+1+1+1+1+0+1+1+1+1+1+1+1+1+1+0+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+1+0+1+1+1+1+1+1+1+0+1+1+1+1+1+1+1+0+0+0+0+1+1+1+0+0 = 47$$

4.2.2.6 *Perimeter*

Perimeter merupakan jumlah piksel pada daerah tepi objek makanan. Untuk melakukan perhitungan *perimeter*, citra biner terlebih dahulu dilakukan deteksi tepi dengan metode *canny edge detection* sehingga menghasilkan citra pada Tabel 4.11.

Tabel 4.11 Perhitungan *Perimeter*

x\y	1	2	3	4	5	6	7	8
1	0	0	1	**1**	1	0	0	0
2	0	1	0	0	0	1	1	0
3	1	0	0	0	0	0	1	0
4	1	0	0	0	0	0	0	1
5	1	0	0	0	0	0	0	1
6	0	1	0	0	0	0	1	0
7	0	1	1	0	0	0	1	0
8	0	0	0	**1**	1	1	0	0

$$P = 0+0+1+1+1+0+0+0+0+1+0+0+0+1+1+0+1+0+0+0+0+1+0+1+0+0+0+0+0+0+1+0+1+0+0+0+0+0+0+1+0+1+0+0+0+0+1+0+0+1+1+0+0+0+1+0+0+0+1+1+1+0+0 = 20$$

4.2.2.7 *Aspect Ratio*

Fitur *aspect ratio* didapatkan dengan menghitung hasil pembagian antara *major axis length* dan *minor axis length* sesuai Persamaan 2.14. Pada langkah sebelumnya telah diketahui nilai *major axis length* yaitu 8 dan nilai *minor axis length* yaitu 8. Berikut contoh perhitungan *aspect ratio*.

$$AR = \frac{8}{8} = 1$$

4.2.2.8 *Roundness*

Perhitungan fitur *roundness* membutuhkan nilai *perimeter* dan nilai *area* yang telah didapat pada langkah sebelumnya. Setelah diketahui nilai *perimeter* yaitu 20 dan nilai area yaitu 47 kemudian dilakukan perhitungan *roundness* menggunakan Persamaan 2.15. Berikut contoh perhitungan *roundness*.

$$R = \frac{4x3{,}14x47}{20^2} = 1{,}48$$

4.2.2.9 *Rectangularity*

Perhitungan fitur *rectangularity* membutuhkan nilai *area*, *major axis length* dan *minor axis length* yang telah didapat pada langkah sebelumnya. Setelah diketahui nilai *area* yaitu 47, nilai *major axis length* yaitu 8 dan *minor axis length* yaitu 8 kemudian dilakukan perhitungan fitur *rectangularity* sesuai Persamaan 2.16. Berikut contoh perhitungan *rectangularity*.

$$N = \frac{47}{8x8} = 0{,}73$$

4.2.2.10 *Compactness*

Perhitungan fitur *compactness* membutuhkan nilai *area* dan nilai *perimeter* yang telah didapat pada langkah sebelumnya. Setelah diketahui nilai *perimeter* yaitu 20 dan nilai *area* yaitu 47 kemudian dilakukan perhitungan fitur *compactness* sesuai Persamaan 2.17. Berikut contoh perhitungan *compactness*.

$$C = \frac{20}{\sqrt{47}} = 2{,}917$$

4.2.2.11 *Narrow Factor*

Perhitungan fitur *narrow factor* membutuhkan nilai *diameter* dan nilai *major axis length* yang telah didapat pada langkah sebelumnya. Setelah diketahui nilai *diameter* yaitu 7,62 dan nilai *major axis length* yaitu 8 kemudian dilakukan perhitungan fitur *narrow factor* sesuai Persamaan 2.18. Berikut contoh perhitungan *narrow factor*.

$$NF = \frac{7{,}62}{8} = 0{,}95$$

4.2.2.12 Rasio *Perimeter* dengan *Diameter*

Perhitungan rasio *perimeter* dengan *diameter* membutuhkan nilai *perimeter* dan nilai *diameter* yang telah didapat pada langkah sebelumnya. Setelah diketahui nilai *perimeter* yaitu 20 dan nilai *diameter* yaitu 7,62 kemudian dilakukan perhitungan fitur rasio perimeter dengan diameter sesuai Persamaan 2.19. Berikut contoh perhitungan fitur rasio *perimeter* dengan *diameter*.

$$P_D = \frac{20}{7{,}62} = 2{,}62$$

4.2.2.13 Rasio *Perimeter* dengan *Major Axis Length*

Perhitungan rasio *perimeter* dengan *major axis length* menggunakan nilai *perimeter* dan *major axis length* yang telah didapat pada langkah sebelumnya. Setelah didapatkan nilai *perimeter* yaitu 20 dan nilai *major axis length* yaitu 8 maka dilakukan perhitungan rasio *perimeter* dengan *major axis length*. Berikut perhitungan fitur rasio *perimeter* dengan *major axis length* sesuai Persamaan 2.20.

$$P_L = \frac{20}{8} = 2{,}5$$

4.2.2.14 Rasio *Perimeter dengan Major Axis Length dan Minor Axis Length*

Perhitungan rasio *perimeter* dengan *major axis length* dan *minor axis length* menggunakan nilai *perimeter*, *major axis length* dan *minor axis length* yang telah didapat pada langkah sebelumnya. Setelah didapatkan nilai *perimeter* yaitu 20, nilai *major axis length* yaitu 8 dan nilai *minor axis length* yaitu 8 maka dilakukan perhitungan rasio *perimeter* dengan *major axis length* dan *minor axis length*. Berikut perhitungan fitur rasio *perimeter* dengan *major axis length* dan *minor axis length* sesuai Persamaan 2.21.

$$P_{LW} = \frac{20}{(8+8)} = 1{,}25$$

4.2.3 Ekstraksi Fitur Warna *Color Moment*

Pada tahap ekstraksi fitur warna *Color Moment* dilakukan perhitungan nilai distribusi warna terhadap setiap citra tersegmentasi yang dihasilkan pada proses sebelumnya. Distribusi warna yang dihitung meliputi *mean*, *standard deviation*, *skewness* dan *kurtosisi*. Perhitungan *Color Moment* menggunakan matriks 3x3 pada Tabel 4.12 yang berasal dari pojok kiri atas citra tersegmentasi Tabel 4.8.

Tabel 4.12 Matriks Perhitungan *Color Moment*

Red			*Green*			*Blue*		
0	0	201	0	0	148	0	0	72
0	199	191	0	151	134	0	88	57
220	173	185	161	116	121	99	62	64

4.2.3.1 *Mean*

Berdasarkan matriks pada Tabel 4.12 maka dilakukan perhitungan *mean* dengan rumus sesuai Persamaan 2.22 sebagai berikut.

$$\mu_1 = \frac{0+0+201+0+199+191+220+173+185}{3x3} = \frac{1169}{9} = 129{,}89$$

$$\mu_2 = \frac{0+0+148+0+151+134+161+116+121}{3x3} = \frac{831}{9} = 92{,}33$$

$$\mu_3 = \frac{0+0+72+0+88+57+99+62+64}{3x3} = \frac{442}{9} = 49{,}11$$

4.2.3.2 *Standard deviation*

Berdasarkan matriks pada Tabel 4.12 maka dilakukan perhitungan *standard deviation* dengan rumus sesuai Persamaan 2.23 sebagai berikut.

$$\sigma_1 = \sqrt{\frac{(0-129{,}89)^2+(0-129{,}89)^2+(201-129{,}89)^2+(0-129{,}89)^2+(199-129{,}89)^2+(191-129{,}89)^2+(220-129{,}89)^2+(173-129{,}89)^2+(185-129{,}89)^2}{3x3}}$$

$$= \sqrt{8577{,}43} = 92{,}61$$

$$\sigma_2 = \sqrt{\frac{(0-92{,}33)^2+(0-92{,}33)^2+(148-92{,}33)^2+(0-92{,}33)^2+(151-92{,}33)^2+(134-92{,}33)^2+(161-92{,}33)^2+(116-92{,}33)^2+(121-92{,}33)^2}{3x3}}$$

$$= \sqrt{4438{,}89} = 66{,}62$$

$$\sigma_3 = \sqrt{\frac{(0-49{,}11)^2+(0-49{,}11)^2+(72-49{,}11)^2+(0-49{,}11)^2+(88-49{,}11)^2+(57-49{,}11)^2+(99-49{,}11)^2+(62-49{,}11)^2+(64-49{,}11)^2}{3x3}}$$

$$= \sqrt{1356{,}76} = 36{,}83$$

4.2.3.3 *Skewness*

Berdasarkan matriks pada Tabel 4.12 maka dilakukan perhitungan *skewness* dengan rumus sesuai Persamaan 2.24 sebagai berikut.

$$\theta_1 = \frac{(0-129{,}89)^3+(0-129{,}89)^3+(201-129{,}89)^3+(0-129{,}89)^3+(199-129{,}89)^3+(191-129{,}89)^3+(220-129{,}89)^3+(173-129{,}89)^3+(185-129{,}89)^3}{3x3x92{,}61^3}$$

$$= \frac{-4676985}{7148520} = \text{-}0{,}654$$

$$\theta_2 = \frac{(0-92{,}33)^3+(0-92{,}33)^3+(148-92{,}33)^3+(0-92{,}33)^3+(151-92{,}33)^3+(134-92{,}33)^3+(161-92{,}33)^3+(116-92{,}33)^3+(121-92{,}33)^3}{3x3x66{,}62^3}$$

$$= \frac{-1554209}{2661071} = \text{-}0{,}584$$

$$\theta_3 = \frac{(0-49{,}11)^3+(0-49{,}11)^3+(72-49{,}11)^3+(0-49{,}11)^3+(88-49{,}11)^3+(57-49{,}11)^3+(99-49{,}11)^3+(62-49{,}11)^3+(64-49{,}11)^3}{3x3x36{,}83^3}$$

$$= \frac{-154447}{449622,1} = \text{-0,3435}$$

4.2.3.4 *Kurtosis*

Berdasarkan matriks pada Tabel 4.12 maka dilakukan perhitungan *kurtosis* dengan rumus sesuai Persamaan 2.25 sebagai berikut.

$$\gamma_1 = \frac{(0-129,89)^4 + (0-129,89)^4 + (201-129,89)^4 + (0-129,89)^4 + (199-129,89)^4 + (191-129,89)^4 + (220-129,89)^4 + (173-129,89)^4 + (185-129,89)^4}{3x3x92,61^4}$$

$$= \frac{994849690,8}{662024476} = 1,5027$$

$$\gamma_2 = \frac{(0-92,33)^4 + (0-92,33)^4 + (148-92,33)^4 + (0-92,33)^4 + (151-92,33)^4 + (134-92,33)^4 + (161-92,33)^4 + (116-92,33)^4 + (121-92,33)^4}{3x3x66,62^4}$$

$$= \frac{265733264,7}{177280522,4} = 1,4989$$

$$\gamma_3 = \frac{(0-49,11)^4 + (0-49,11)^4 + (72-49,11)^4 + (0-49,11)^4 + (88-49,11)^4 + (57-49,11)^4 + (99-49,11)^4 + (62-49,11)^4 + (64-49,11)^4}{3x3x36,83^4}$$

$$= \frac{26288706}{16559583} = 1,587522$$

4.2.4 Perhitungan Normalisasi Fitur

Normalisasi fitur digunakan untuk membuat data memiliki rentang nilai yang sama. Metode yang digunakan adalah *min-max normalization*. Pada contoh perhitungan normalisasi ini data yang digunakan adalah 10 citra dengan rincian 3 citra donat, 3 citra roti gandum dan 4 citra roti tawar. Perhitungan normalisasi fitur dimulai dengan cara mencari nilai minimum dan maksimum dari setiap fitur. Misalnya pada Tabel 4.14 untuk fitur L (*major axis length*) didapatkan nilai maksimum sebesar 419 dan minimum sebesar 305. Kemudian dihitung nilai hasil normalisasi menggunakan Persamaan 2.26. Contoh perhitungan normalisasi untuk fitur major axis length citra donat_1 adalah sebagai berikut.

$$s' = \frac{306-305}{419-305} = \frac{1}{114} = 0,01$$

Perhitungan normalisasi dilakukan untuk semua fitur pada setiap data hasil ekstraksi fitur. Hasil lengkap perhitungan nilai normalisasi dapat dilihat pada Tabel 4.15.

4.2.5 Perhitungan Pengukuran Kemiripan

Pada tahap ini dilakukan perhitungan untuk mengukur kemiripan antara *query* dengan citra pada basis data. Pengukuran kemiripan dilakukan dengan cara menghitung jarak antara citra *query* dengan citra pada basis data dengan menggunakan *euclidean distance* sesuai rumus pada Persamaan 2.27.

Tabel 4.16 merupakan citra *query* sebagai masukan pada sistem pencarian. Tabel 4.17 merupakan data *corpus* citra sebagai hasil pencarian. Contoh perhitungan *euclidean distance* antara citra *query* dengan *corpus* citra donat_1 adalah sebagai berikut.

$$di = \sqrt{\begin{matrix}(0{,}0-0{,}73)^2+(0{,}0-0{,}7)^2+(0{,}0-0{,}207)^2+(0{,}0-0{,}67)^2 \\ +(0{,}0-0{,}808)^2+(0{,}0-0{,}54)^2+(0{,}0-0{,}905)^2+(0{,}1-0{,}72)^2 \\ +(1{,}0-0{,}16)^2+(1{,}0-0{,}18)^2+(0{,}0-0{,}81)^2+(0{,}97-0{,}05)^2 \\ +(0{,}0-0{,}88)^2+(0{,}94-0{,}62)^2+(0{,}65-0{,}89)^2+(0{,}825-0{,}124)^2 \\ +(0{,}925-0{,}13)^2+(0{,}95-0{,}76)^2+(0{,}18-0{,}0)^2+(0{,}0-0{,}82)^2 \\ +(0{,}0-0{,}92)^2+(0{,}02-0{,}81)^2+(0{,}01-0{,}85)^2+(0{,}0-0{,}80)^2 \\ +(0{,}93-0{,}63)^2+(0{,}98-0{,}13)^2+(1{,}0-0{,}08)^2\end{matrix}} = \sqrt{13{,}7966} = 3{,}714$$

Setelah dilakukan perhitungan jarak tahap selanjutnya yaitu mengurutkan dari nilai terkecil sampai terbesar. Hasil pengurutan nilai jarak dapat dilihat pada Tabel 4.13.

Tabel 4.13 Hasil Pengukuran Kemiripan

***Corpus* citra**	**Euclidean distance**	**Keterangan**
tawar_2	0,769584	Relevan
tawar_3	1,744366	Relevan
tawar_1	1,935771	Relevan
donat_2	3,311519	Tidak Relevan
gandum_1	3,519541	Tidak Relevan
gandum_2	3,527428	Tidak Relevan
gandum_3	3,570643	Tidak Relevan
donat_1	3,71438	Tidak Relevan
donat_3	4,13478	Tidak Relevan

Tabel 4.14 Contoh Data Untuk Pehitungan Normalisasi Fitur

Data	L	W	D	Cnt_x	Cnt_y	A	P	AR	R	N	C	NF	PD	PL	PLW	X1	X2	X3	δ1	δ1	δ3	S1	S2	S3	K1	K2	K3
donat_1	423	416	451.8	213	209	133444	1641	1.02	0.62	0.76	4.49	1.07	3.63	3.88	1.96	46.9	87.7	132.9	37.4	57.8	79.2	0.49	-0.36	-0.90	2.6	2.0	2.14
donat_2	424	420	465.2	213	209	144585	1577	1.01	0.73	0.81	4.15	1.10	3.39	3.72	1.87	45.9	88.4	135.4	38.7	55.2	72.3	0.62	-0.29	-0.99	2.6	2.1	2.57
donat_3	431	421	451.4	219	210	134321	1690	1.02	0.59	0.74	4.61	1.05	3.74	3.92	1.98	39.1	83.7	130.0	37.7	59.4	81.7	0.85	-0.15	-0.75	2.9	1.9	1.95
gandum_1	472	437	583.4	245	210	173880	1678	1.08	0.78	0.84	4.02	1.24	2.88	3.56	1.85	101.8	113.3	125.9	57.7	59.6	61.4	-0.56	-0.81	-1.13	2.2	2.5	3.05
gandum_2	387	453	501.8	197	203	127310	1456	0.85	0.75	0.73	4.08	1.30	2.90	3.76	1.73	71.4	81.5	95.8	53.4	57.7	63.0	-0.17	-0.35	-0.64	1.6	1.6	1.80
gandum_3	473	438	579.9	244	210	172931	1669	1.08	0.78	0.83	4.01	1.23	2.88	3.53	1.83	93.1	102.7	115.8	55.7	57.0	59.3	-0.40	-0.65	-0.96	2.1	2.3	2.78
tawar_1	376	434	543.2	189	219	145382	1531	0.87	0.78	0.89	4.02	1.44	2.82	4.07	1.89	96.0	126.0	138.2	43.8	51.5	52.6	-1.05	-1.51	-1.86	3.2	4.2	5.36
tawar_2	325	375	473.7	158	182	109909	1330	0.87	0.78	0.90	4.01	1.46	2.81	4.09	1.90	95.2	129.0	144.6	42.4	50.5	52.3	-1.06	-1.57	-1.97	3.3	4.5	5.92
tawar_3	369	423	531.2	186	213	139063	1496	0.87	0.78	0.89	4.01	1.44	2.82	4.05	1.89	95.9	125.9	139.6	43.9	51.6	53.2	-1.04	-1.49	-1.86	3.1	4.2	5.36
tawar_4	289	330	417.5	148	167	85990	1173	0.88	0.79	0.90	4.00	1.44	2.81	4.06	1.89	90.8	125.4	141.9	41.1	49.4	51.3	-1.02	-1.55	-1.97	3.1	4.5	5.94
MAX	473	453	583.4	245	219	173880	1690	1.08	0.79	0.90	4.61	1.46	3.74	4.09	1.98	101.8	129.0	144.6	57.7	59.6	81.7	0.85	-0.15	-0.64	3.3	4.5	5.94
MIN	289	330	417.5	148	167	85990	1173	0.85	0.59	0.73	4.00	1.05	2.81	3.53	1.73	39.1	81.5	95.8	37.4	49.4	51.3	-1.06	-1.57	-1.97	1.6	1.6	1.80

Tabel 4.15 Perhitungan Hasil Normalisasi Fitur

data	L	W	D	Cnt_x	Cnt_y	A	P	AR	R	N	C	NF	PD	PL	PLW	X1	X2	X3	δ1	δ1	δ3	S1	S2	S3	K1	K2	K3
donat_1	0.73	0.70	0.207	0.67	0.808	0.540	0.905	0.72	0.16	0.18	0.81	0.05	0.88	0.62	0.89	0.124	0.130	0.76	0.00	0.82	0.92	0.81	0.85	0.80	0.63	0.13	0.08
donat_2	0.73	0.73	0.287	0.67	0.808	0.667	0.781	0.69	0.72	0.49	0.24	0.12	0.62	0.34	0.54	0.109	0.145	0.81	0.07	0.57	0.69	0.88	0.90	0.74	0.61	0.18	0.19
donat_3	0.77	0.74	0.204	0.73	0.827	0.550	1.000	0.75	0.00	0.08	1.00	0.00	1.00	0.70	1.00	0.000	0.046	0.70	0.01	0.98	1.00	1.00	1.00	0.91	0.80	0.10	0.04
gandum_1	0.99	0.87	1.000	1.00	0.827	1.000	0.977	1.00	0.95	0.67	0.04	0.46	0.07	0.05	0.45	1.000	0.671	0.62	1.00	1.00	0.33	0.26	0.53	0.63	0.36	0.31	0.30
gandum_2	0.53	1.00	0.508	0.51	0.692	0.470	0.547	0.00	0.84	0.00	0.13	0.61	0.10	0.41	0.00	0.515	0.000	0.00	0.79	0.81	0.38	0.47	0.86	1.00	0.00	0.00	0.00
gandum_3	1.00	0.88	0.979	0.99	0.827	0.989	0.959	1.00	0.97	0.62	0.02	0.44	0.08	0.00	0.39	0.862	0.447	0.41	0.90	0.74	0.26	0.34	0.65	0.76	0.30	0.24	0.24
tawar_1	0.47	0.85	0.758	0.42	1.000	0.676	0.692	0.05	0.97	0.94	0.02	0.97	0.01	0.96	0.63	0.907	0.938	0.87	0.32	0.21	0.04	0.00	0.04	0.09	0.94	0.89	0.86

tawar_2	0.20	0.37	0.339	0.10	0.288	0.272	0.304	0.05	0.98	1.00	0.02	1.00	0.00	1.00	0.67	0.895	1.000	1.00	0.25	0.11	0.03	0.00	0.00	0.00	1.00	1.00	1.00
tawar_3	0.43	0.76	0.685	0.39	0.885	0.604	0.625	0.08	0.98	0.94	0.02	0.96	0.01	0.93	0.62	0.905	0.936	0.90	0.32	0.22	0.06	0.01	0.05	0.09	0.93	0.89	0.86
tawar_4	0.00	0.00	0.000	0.00	0.000	0.000	0.000	0.10	1.00	1.00	0.00	0.97	0.00	0.94	0.65	0.825	0.925	0.95	0.18	0.00	0.00	0.02	0.01	0.00	0.93	0.98	1.00

Tabel 4.16 Contoh Data Uji

Data	L	W	D	Cnt_x	Cnt_y	A	P	AR	R	N	C	NF	PD	PL	PLW	X1	X2	X3	δ1	δ1	δ3	S1	S2	S3	K1	K2	K3
tawar_4	0.00	0.00	0.000	0.00	0.000	0.000	0.000	0.10	1.00	1.00	0.00	0.97	0.00	0.94	0.65	0.825	0.925	0.95	0.18	0.00	0.00	0.02	0.01	0.00	0.93	0.98	1.00

Tabel 4.17 Contoh *Corpus* Citra

Data	L	W	D	Cnt_x	Cnt_y	A	P	AR	R	N	C	NF	PD	PL	PLW	X1	X2	X3	δ1	δ1	δ3	S1	S2	S3	K1	K2	K3
donat_1	0.73	0.70	0.207	0.67	0.808	0.540	0.905	0.72	0.16	0.18	0.81	0.05	0.88	0.62	0.89	0.124	0.130	0.76	0.00	0.82	0.92	0.81	0.85	0.80	0.63	0.13	0.08
donat_2	0.73	0.73	0.287	0.67	0.808	0.667	0.781	0.69	0.72	0.49	0.24	0.12	0.62	0.34	0.54	0.109	0.145	0.81	0.07	0.57	0.69	0.88	0.90	0.74	0.61	0.18	0.19
donat_3	0.77	0.74	0.204	0.73	0.827	0.550	1.000	0.75	0.00	0.08	1.00	0.00	1.00	0.70	1.00	0.000	0.046	0.70	0.01	0.98	1.00	1.00	1.00	0.91	0.80	0.10	0.04
gandum_1	0.99	0.87	1.000	1.00	0.827	1.000	0.977	1.00	0.95	0.67	0.04	0.46	0.07	0.05	0.45	1.000	0.671	0.62	1.00	1.00	0.33	0.26	0.53	0.63	0.36	0.31	0.30
gandum_2	0.53	1.00	0.508	0.51	0.692	0.470	0.547	0.00	0.84	0.00	0.13	0.61	0.10	0.41	0.00	0.515	0.000	0.00	0.79	0.81	0.38	0.47	0.86	1.00	0.00	0.00	0.00
gandum_3	1.00	0.88	0.979	0.99	0.827	0.989	0.959	1.00	0.97	0.62	0.02	0.44	0.08	0.00	0.39	0.862	0.447	0.41	0.90	0.74	0.26	0.34	0.65	0.76	0.30	0.24	0.24
tawar_1	0.47	0.85	0.758	0.42	1.000	0.676	0.692	0.05	0.97	0.94	0.02	0.97	0.01	0.96	0.63	0.907	0.938	0.87	0.32	0.21	0.04	0.00	0.04	0.09	0.94	0.89	0.86
tawar_2	0.20	0.37	0.339	0.10	0.288	0.272	0.304	0.05	0.98	1.00	0.02	1.00	0.00	1.00	0.67	0.895	1.000	1.00	0.25	0.11	0.03	0.00	0.00	0.00	1.00	1.00	1.00
tawar_3	0.43	0.76	0.685	0.39	0.885	0.604	0.625	0.08	0.98	0.94	0.02	0.96	0.01	0.93	0.62	0.905	0.936	0.90	0.32	0.22	0.06	0.01	0.05	0.09	0.93	0.89	0.86

4.2.6 Pengujian MAP

Berdasarkan hasil pengukuran kemiripan pada Tabel 4.13. Jika dicitra yang di *retrieve* sebanyak 5 maka akan didapatkan hasil pencarian yang ditunjukkan pada Tabel 4.18.

Tabel 4.18 Hasil Pencarian

***Corpus* Citra**	**Keterangan**
tawar_3	Relevan
tawar_2	Relevan
tawar_1	Relevan
donat_2	Tidak Relevan
gandum_1	Tidak Relevan

Pengujian dilakukan dengan menghitung nilai MAP yang didapat dengan cara menghitung nilai rata-rata *precision* pada setiap citra relevan yang di-*retrieve*. Nilai *precision* untuk setiap citra relevan yang di-*retrieve* ditunjukkan pada Tabel 4.19.

Tabel 4.19 Nilai *Precision*

***Corpus* Citra**	**Keterangan**	***Precision***
tawar_3	Relevan	1
tawar_2	Relevan	1
tawar_1	Relevan	1
donat_2	Tidak Relevan	0
gandum_1	Tidak Relevan	0

Dari Tabel 4.19 dilakukan perhitungan nilai MAP sesuai pada Persamaan 2.32 sebagai berikut.

$$MAP = \frac{1+1+1}{3} = 1$$

4.3 Perancangan Pengujian dan Evaluasi

Setelah sistem diimplementasikan, dilakukan tahap pengujian dan evaluasi terhadap hasil pencarian. Pengujian dilakukan pada dataset dengan rasio perbandingan 80% sebagai *corpus* citra dan 20% untuk citra *query*.

Pengujian pertama dilakukan dengan mengukur nilai MAP pada nilai k-rank yang bervariasi. Nilai *k-rank* yang digunakan antara lain 10, 25, 50, 75 dan 100. Pengujian ini bertujuan untuk mengetahui nilai *k-rank* yang tepat sehingga menghasilkan nilai MAP paling optimal. Skenario pengujian dapat ditunjukkan pada Tabel 4.20.

Tabel 4.20 Skenario Pengujian *k-rank*

k-rank	**100**	**75**	**50**	**25**	**10**
MAP					

Pengujian kedua dilakukan dengan menguji penggunaan fitur *Simple Morphological Shape Descriptors* dan *Color Moment*. Pada setiap pengujian akan dikalkulasi nilai MAP. Skenario pengujian terhadap kombinasi fitur *Simple Morphological Shape Descriptors* dan *Color Moment* dapat dilihat pada Tabel 4.21.

Tabel 4.21 Skenario Pengujian Kombinasi Fitur

Kombinasi Fitur	***Simple Morphological Shape Descriptors***	***Color Moment***	***Simple Morphological Shape Descriptors* dan *Color Moment***
MAP			

BAB 5 IMPLEMENTASI

5.1 Spesifikasi Sistem

Sistem merupakan suatu representasi dari gabungan antara kebutuhan perangkat keras dan kebutuhan perangkat lunakn untuk melakukan implementasi. Spesifikasi sistem diimplementasikan pada spesifikasi perangkat keras (*hardware*) dan spesifikasi perangkat lunak (*software*).

5.1.1 Spesifikasi Perangkat Keras

Pembangunan sistem pencarian resep makanan berdasarkan citra makanan didukung oleh perangkat keras komputer dengan spesifikasi untuk setiap komponennya ditunjukkan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Perangkat Keras

Nama Komponen	Spesifikasi
Processor	AMD A8-7410 APU with AMD Radeon R5 Graphics 2.20 GHz
RAM	8.00 GB
Harddisk	500 GB

5.1.2 Spesifikasi Perangkat Lunak

Pembangunan sistem pencarian resep makanan berdasarkan citra makanan didukung oleh perangkat keras komputer dengan spesifikasi untuk setiap komponennya ditunjukkan pada Tabel 5.2.

Tabel 5.2 Spesifikasi Perangkat Lunak

Nama Komponen	Spesifikasi
Sistem Operasi	Windows 10 Pro
Bahasa Pemrograman	Python 2
Aplikasi Pemrograman	JetBrains Pycharm Community Edition 2017.2.2

5.2 Implementasi Program

Implementasi program menjelaskan tahapan proses sistem pencarian resep makanan berdasarkan citra makanan menggunakan ekstraksi fitur *Simple Morphological Shape Descriptors* dan *Color Moment.* Secara garis besar, proses utama pada sistem ini meliputi proses *pre-processing*, ekstraksi fitur *Simple Morphological Shape Descriptors,* ekstrakfi fitur *Color Moment*, proses normalisasi fitur dan proses pengukuran kemiripan menggunakan *Euclidean Distance*. Fungsi-fungsi yang digunakan dijabarkan pada Tabel 5.3.

Tabel 5.3 Daftar Fungsi Implementasi Program

No.	Proses	Fungsi	Keterangan
1.	*Pre-processing*	`preprocessing()`	Fungsi untuk memanggil semua fungsi yang ada di proses *pre-processing*.
		`resize_citra()`	Fungsi untuk melaukan resize citra makanan
		`filtering()`	Fungsi untuk melakukan filtering citra makanan
		`konversi()`	Fungsi untuk mengubah color space RGB ke L*a*b
		`Binerisasi()`	Fungsi untuk melakukan binerisasi citra
		`cropping()`	Fungsi untuk memotong citra
		`segmentasi()`	Fungsi untuk melakukan segmentasi citra makanan
2.	Ekstraksi fitur *Simple Morphological Shape Descriptors*	`Smsd()`	Fungsi untuk memanggil semua fungsi yang ada pada proses ekstraksi fitur Simple Morphological Shape Descriptors
		`Major_length()`	Fungsi untuk menghitung fitur major axis length
		`Minor_length()`	Fungsi untuk menghitung fitur minor axis length
		`Diameter()`	Fungsi untuk menghitung fitur diameter
		`Centroid()`	Fungsi untuk menghitung fitur centroid
		`Area()`	Fungsi untuk menghitung fitur area
		`Perimeter()`	Fungsi untuk menghitung fitur perimeter
		`Aspect_Ratio()`	Fungsi untuk menghitung fitur aspect ratio
		`Roundness()`	Fungsi untuk menghitung fitur roundness
		`Rectangularity()`	Fungsi untuk menghitung fitur rectangularity
		`Compactness()`	Fungsi untuk menghitung compactness

		`Narrow_factor()`	Fungsi untuk menghitung narrow factor
		`Ratio_PD()`	Fungsi untuk menghitung fitur rasio perimeter dengan diameter
		`Ratio_PL()`	Fungsi untuk menghitung fitur rasio perimeter dengan major axis length
		`Ratio_PLW()`	Fungsi untuk menghitung fitur rasio perimeter dengan major dan minor axis length
3.	Ekstraksi Fitur *Color Moment*	`Color_moment()`	Fungsi untuk memanggil semua fungsi yang ada pada proses ekstraksi fitur Color Moment
		`Mean()`	Fungsi untuk menghitung fitur *mean*
		`Standar_Deviasi()`	Fungsi untuk menghitung fitur *standard deviation*
		`Skewness()`	Fungsi untuk menghitung fitur skewness
		`Kurtosis()`	Fungsi untuk menghitung fitur kurtosis
4.	Normalisasi	`Normalisasi()`	Fungsi untuk melakukan normalisasi semua fitur
5.	Pengukuran Kemiripan *Euclidean Distance*	`Euclidean_distance()`	Fungsi untuk menghitung nilai kedekatan antara *query* pencarian dengan citra pada basis data

5.2.1 Implementasi *Pre-processing*

Pada implementasi *pre-processing* dilakukan pemanggilan seluruh fungsi yang terdapat pada proses *pre-processing*. Fungsi-fungsi tersebut terdiri dari *filtering*, konversi RGB ke L*a*b, binerisasi, *cropping* dan segmentasi citra. Proses tersebut dapat dilihat pada Kode Program 5.1.

```
1  def preprocessing(self, img):
2      img = self.resize_citra(img)
3      citra_blur = self.filtering(img)
4      b = self.konversi(citra_blur)
5      citra_biner = self.binerisasi(b)
6      crop = self.cropping(citra_biner,img)
7      citra_tersegmentasi = self.segmentasi(crop[0], crop[1])
8      return (crop[0],citra_tersegmentasi)
```

Kode Program 5.1 *Pre-processing*

Penjelasan dari Kode Program 5.1 *Pre-processing* adalah sebagai berikut.

Baris 2 : memanggil fungsi resize citra kemudian menyimpan hasilnya pada variabel img.
Baris 3 : memanggil fungsi filtering kemudian menyimpan hasilnya pada variabel citra_blur.
Baris 4 : memanggil fungsi konversi kemudian menyimpan hasilnya pada variabel b.
Baris 5 : memanggil fungsi binerisasi kemudian menyimpan hasilnya pada variabel citra_biner.
Baris 6 : memanggil fungsi cropping kemudian menyimpan hasilnya pada variabel crop.
Baris 7 : memanggil fungsi segmentasi kemudian menyimpan hasilnya pada variabel citra_tersegmentasi.

5.2.1.1 Implementasi Proses *Resize* Citra

Implementasi proses *resize* citra dilakukan dengan mengubah ukuran citra menjadi 500x500 piksel. Pada penelitian ini proses *resize* dilakukan dengan cara memanggil fungsi *resize* yang terdapat pada Open CV. Implementasi proses *filtering* dapat dilihat pada Kode Program 5.3.

```
1  def resize_citra(self, img):
2      img_resize = cv2.resize(img, (500, 500))
3      return img_resize
```

Kode Program 5.2 Proses *Resize* Citra

Penjelasan dari Kode Program 5.2 Proses *Resize* Citra adalah sebagai berikut.

Baris 2 : memanggil fungsi resize dari library open CV.

5.2.1.2 Implementasi Proses *Filtering*

Implementasi proses *filtering* menggunakan *Gaussian Filter*. Pada penelitian ini proses *filtering* dilakukan dengan cara memanggil fungsi *GaussianBlur* yang terdapat pada Open CV. Kernel yang digunakan pada proses *gaussian filter* berukuran 11x11. Implementasi proses *filtering* dapat dilihat pada Kode Program 5.3.

```
1  def filtering(self,citra_RGB):
2      citra_blur = cv2.GaussianBlur(citra_RGB, (11, 11), 0)
3      return citra_blur
```

Kode Program 5.3 Proses *Filtering*

Penjelasan dari Kode Program 5.3 Proses *Filtering* adalah sebagai berikut.

Baris 2 : memanggil fungsi GaussianBlur dari library open CV.

5.2.1.3 Implementasi Proses konversi RGB ke L*a*b

Implementasi proses konversi RGB ke L*a*b pada penelitian ini dilakukan dengan cara memanggil fungsi cvtColor yang terdapat pada *library* OpenCV. Setelah didapat hasil konversinya kemudian diambil matriks ke-2 yaitu komponen

b. Implementasi proses konversi RGB ke L*a*b dapat dilihat pada Kode Program 5.4.

```
1   def konversi(self,citra_blur):
2       lab = cv2.cvtColor(citra_blur, cv2.COLOR_BGR2LAB)
3       b = lab[:, :, 2]
4       return b
```

Kode Program 5.4 Proses konversi RGB ke L*a*b

Penjelasan dari Kode Program 5.4 Proses konversi RGB ke L*a*b adalah sebagai berikut.
Baris 2 : untuk melakukan konversi RGB ke L*a*b.
Baris 3 : untuk mengambil matriks komponen b pada citra L*a*b.

5.2.1.4 Implementasi Proses Binerisasi

Implementasi proses binerisasi dimulai dengan melakukan *thresholding* terhadap komponen b citra L*a*b hasil proses sebelumnya. Proses *thresholding* dilakukan dengan metode *Otsu Thresholding* dengan memanfaatkan *library* OpenCV. Setelah itu dilakukan operasi morfologi *closing* untuk memperbaiki citra hasil *thresholding*. Implementasi proses binerisasi dapat dilihat pada Kode Program 5.5.

```
1   def binerisasi(self,b):
2       ret3, th3 = cv2.threshold(b, 0, 255, cv2.THRESH_BINARY
    + cv2.THRESH_OTSU)
3       kernel = np.ones((3, 3), np.uint8)
4       dilation = cv2.dilate(inv, kernel, iterations=1)
5       erotion = cv2.erode(dilation, kernel, iterations=1)
6       erotion[erotion>0]=1
7       return erotion
```

Kode Program 5.5 Proses Binerisasi

Penjelasan dari Kode Program 5.5 Proses Binerisasi adalah sebagai berikut.
Baris 2 : untuk melakukan thresholding dengan metode Otsu.
Baris 4 : inisialisasi kernel berukuran 3x3.
Baris 5 : untuk melakukan dilasi sebanyak 1 kali iterasi
Baris 6 : untuk melakukan erosi sebnayak 1 kali iterasi

5.2.1.5 Implementasi Proses *Cropping*

Implementasi proses cropping dilakukan dengan mencari semua contour pada citra biner. Setelah itu dicari contour terbesar dengan cara membandingkan setiap contour pada daftar. Contour terbesar tersebut digunakan untuk mendapatkan nilai maksimum koordinat. Kemudian dilakukan cropping untuk citra biner dan citra RGB berdasarkan nilai maksimum koordinat yang telah didapatkan. Implementasi proses cropping dapat dilihat pada Kode Program 5.6.

```
1   def cropping(self,citra_biner, citra_RGB):
2       im2, contours, hierarchy =
    cv2.findContours(citra_biner, cv2.RETR_TREE,
    cv2.CHAIN_APPROX_SIMPLE)
3       max = 0
4       for x in range(len(contours)):
```

```
5           if (len(contours[x]) > max):
6               max = len(contours[x])
7               cnt = contours[x]
8       leftmost = tuple(cnt[cnt[:, :, 0].argmin()][0])
9       rightmost = tuple(cnt[cnt[:, :, 0].argmax()][0])
10      topmost = tuple(cnt[cnt[:, :, 1].argmin()][0])
11      bottommost = tuple(cnt[cnt[:, :, 1].argmax()][0])
12      cropBW = citra_biner[topmost[1]:bottommost[1] + 1,
    leftmost[0]:rightmost[0] + 1]
13      cropImage = citra_RGB[topmost[1]:bottommost[1] + 1,
    leftmost[0]:rightmost[0] + 1]
14      return (cropBW,cropImage)
```

Kode Program 5.6 Proses *Cropping*

Penjelasan dari Kode Program 5.6 Proses *Cropping* adalah sebagai berikut.

Baris 2 : mendapatkan semua contour pada citra biner dengan menggunakan method findContour milik library OpenCV

Baris 3 : inisialisasi variabel max untuk menampung panjang array maksimum

Baris 4-7 : perulangan untuk mendapatkan contour terbesar dengan cara membandingkan nilai variabel max dengan panjang array contour ke-x

Baris 5-11 : mendapatkan leftmost, rightmost, topmost dan bottommost dari array cnt

Baris 12-13 : memotong citra biner dan citra RGB

5.2.1.6 Implementasi Proses Segmentasi Citra

Impelemntasi proses segmentasi citra dilakukan dengan cara mengalikan setiap piksel pada citra biner dengan citra RGB. Hasilnya akan didapatkan citra RGB yang memiliki *background* berwarna hitam. Proses segmentasi citra dapat dilihat pada Kode Program 5.7.

```
1   def segmentasi(self, bw, cropImage):
2       bw[bw>0]=1
3       h, w = bw.shape[:2]
4       for x in range(h):
5           for y in range(w):
6               cropImage[x][y] = bw[x][y] * cropImage[x][y]
7       return cropImage
```

Kode Program 5.7 Proses Segmentasi Citra

Penjelasan dari Kode Program 5.7 Proses Segmentasi Citra adalah sebagai berikut.

Baris 2 : mengubah citra sehingga setiap pikselnya bernilai 0 atau 1

Baris 3 : menyimpan ukuran panjang dan lebar citra

Baris 4-6 : perulangan untuk mengalikan setiap piksel pada citra biner dengan citra RGB

5.2.2 Implementasi Proses Ekstraksi Fitur *Simple Morphological Shape Descriptors*

Pada implementasi proses ektraksi fitur *simple morphological shape descriptors* dilakukan pemanggilan seluruh fungsi yang terdapat pada proses

tersebut. Fungsi-fungsi tersebut terdiri dari major length, minor length, diameter, centroid, area, perimeter, aspect ratio, roundness, rectangularity, compactness, narrow factor, ratio PD, ratio PL dan ratio PLW. Proses tersebut dapat dilihat pada Kode Program 5.8.

```
1   def smsd(self, img_bw):
2       L = self.major_length(img_bw)
3       W = self.minor_length(img_bw)
4       D = self.diameter(img_bw)
5       Cnt = self.centroid(img_bw)
6       A = self.area(img_bw)
7       P = self.perimeter(img_bw)
8       AR = self.aspect_ratio(L, W)
9       R = self.roundness(A, P)
10      N = self.rectangularity(A, L, W)
11      C = self.compactness(A, P)
12      NF = self.narrow_factor(D, L)
13      PD = self.ratio_PD(P, D)
14      PL = self.ratio_PL(P, L)
15      PLW = self.ratio_PLW(P, L, W)
16      return (L, W, D, Cnt[0], Cnt[1], A, P, AR, R, N, C, NF,
PD, PL, PLW)
```

Kode Program 5.8 Proses Ekstraksi Fitur *Simple Morphological Shape Descriptors*

Penjelasan dari Kode Program 5.8 Proses Ekstraksi Fitur *Simple Morphological Shape Descriptors* adalah sebagai berikut.

Baris 2-16 : memanggil method/fungsi untuk menghitung masing-masing fitur pada ekstraksi fitur bentuk

5.2.2.1 Implementasi Proses Hitung *Major Axis Length*

Pada proses hitung *major axis length* dilakukan pengukuran panjang dan lebar citra biner. Setelah itu nilai panjang dikembalikan sebagai hasil dari proses hitung *major axis length*. Proses hitung *major axis length* dapat dilihat pada Kode Program 5.9.

```
1   def major_length(self, img_bw):
2       L,W = img_bw.shape[:2]
3       return L
```

Kode Program 5.9 Proses Hitung *Major Axis Length*

Penjelasan dari Kode Program 5.9 Proses Hitung *Major Axis Length* adalah sebagai berikut.

Baris 2 : menyimpan ukuran panjang dan lebar citra img_bw

5.2.2.2 Implementasi Proses Hitung *Minor Axis Length*

Pada proses hitung minor axis length dilakukan pengukuran panjang dan lebar citra biner. Setelah itu nilai lebar dikembalikan sebagai hasil dari proses hitung *minor axis length*. Proses hitung *minor axis length* dapat dilihat pada Kode Program 5.9.

```
1   def minor_length(self, img_bw):
2       L, W = img_bw.shape[:2]
3       return W
```

Kode Program 5.10 Proses Hitung *Minor Axis Length*

Penjelasan dari Kode Program 5.10 Proses Hitung *Minor Axis Length* adalah sebagai berikut.

Baris 2 : menyimpan ukuran panjang dan lebar citra img_bw

5.2.2.3 Implementasi Proses Hitung Diameter

Implementasi proses hitung diameter dimulai dengan memperlebar dan memperpanjang citra biner. Hasilnya kemudian dilakukan deteksi tepi dengan metode *canny edge detector* untuk mendapatkan tepi objek makanan. Setelah itu dilakukan penyimpanan setiap koordinat pada piksel yang bernilai 1. Kemudian dilakukan perhitungan jarak antar koordinat. Kemudian dilakukan pengurutan nilai jarak dari terkecil sampai terbesar. Nilai jarak terbesar akan dikembalikan sebagai diameter. Proses hitung diameter dapat dilihat pada Kode Program 5.11.

```
1   Def diameter(self, img_bw):
2       h, w = img_bw.shape[:2]
3       bw = np.zeros((h + 2, w + 2), np.uint8)
4       for x in range(h + 2):
5           for y in range(w + 2):
6               if ((x > 0 and y > 0) and (x < h + 1 and y < w
    + 1)):
7                   bw[x][y] = img_bw[x - 1][y - 1]
8       edges = cv2.Canny(bw, 0, 1)
9       edges[edges > 0] = 1
10      koordinat = []
11      for i in range(h+2):
12          for j in range(w+2):
13              if (edges[i][j] == 1):
14                  koordinat.append((i, j))
15      jarak = []
16      for a in range(len(koordinat)):
17          for b in range(a, len(koordinat)):
18              dist = math.sqrt((koordinat[a][0] -
    koordinat[b][0])**2+(koordinat[a][1]-koordinat[b][1]) ** 2)
19              jarak.append(dist)
20      jarak.sort(reverse=True)
21      D = jarak[0]
22      return D
```

Kode Program 5.11 Proses Hitung Diameter

Penjelasan dari Kode Program 5.11 Proses Hitung Diameter adalah sebagai berikut.

Baris 2 : menyimpan panjang dan lebar citra

Baris 3 : inisialisasi array bernilai 0 yang berukuran 2 piksel lebih panjang dan 2 piksel lebih lebar

Baris 4-7 : perulangan untuk memindah nilai citra img_bw ke array bw yang ukurannya lebih lebar

Baris 8 : deteksi tepi dengan canny

Baris 9 : mengubah citra hasil deteksi tepi sehingga bernilai 1 atau 0

Baris 10 : deklarasi array koordinat untuk menampung koordinat yang pikselnya bernilai 1
Baris 11-14 : perulangan untuk menyimpan koordinat yang pikselnya bernilai 1
Baris 15 : deklarasi array jarak untuk menampung nilai jarak antar koordinat
Baris 16-19 : perulangan untuk menghitung jarak antar koordinat kemudian menyimpanya
Baris 20 : mengurutkan array jarak secara descending

5.2.2.4 Implementasi Proses Hitung *Centroid*

Implementasi proses hitung centroid dilakukan dengan menentukan nilai koordinat titik tengah objek makanan pada citra hitam putih. Hal tersebut dilakukan dengan cara meghitung rata-rata nilai koordinat piksel yang bernilai 1. Hasil dari proses ini menghasilkan dua nilai yaitu koordinat x dan koordinat y. Implementasi proses hitung centroid ditunjukkan pada Kode Program 5.1.

```
1   def centroid(self, img_bw):
2       x = y = 0
3       h, w = img_bw.shape[:2]
4       count = 0
5       for a in range(h):
6           for b in range(w):
7               if (img_bw[a][b] == 1):
8                   count += 1
9                   x += a
10                  y += b
11      x = x / count
12      y = y / count
13      return (x,y)
```

Kode Program 5.12 Proses Hitung *Centroid*

Penjelasan dari Kode Program 5.12 Proses Hitung *Centroid* adalah sebagai berikut.
Baris 2 : inisialisasi variabel x dan y
Baris 3 : menyimpan nilai ukuran citra img_bw ke dalam variabel h dan w
Baris 4 : inisialisasi variabel count bernilai 0
Baris 5-10 : perulangan untuk menghitung jumlah koordinat yang bernilai 1 berdasarkan koordinat x dan y
Baris 11-12 : menghitung nilai koordinat titik tengah

5.2.2.5 Implementasi Proses Hitung Area

Implementasi proses hitung area dilakukan dengan cara mendapatkan luas dari objek makanan pada citra hitam putih. proses mendapatkan luas dilakukan dengan menjumlahkan semua piksel pada citra hitam putih yang bernilai 1. Implementasi proses hitung area ditunjukkan pada Kode Program 5.13.

```
1   def area(self, img):
2       h, w = img.shape[:2]
3       A = 0
4       for x in range(h):
5           for y in range(w):
6
```

```
7                   A += img[x][y]
        return A
```

Kode Program 5.13 Proses Hitung Area

Penjelasan dari Kode Program 5.13 Proses Hitung Area adalah sebagai berikut.

Baris 2 : menyimpan nilai ukuran citra img ke dalam variabel h dan w
Baris 3 : inisialisasi variabel A bernilai 0 untuk menampung nilai area
Baris 4-6 : perulangan untuk menghitung nilai A dengan cara menjumlahkan setiap piksel pada citra img

5.2.2.6 Implementasi Proses Hitung Perimeter

Implementasi proses hitung perimeter dimulai dengan memperlebar dan memperpanjang citra biner. Hasilnya kemudian dilakukan deteksi tepi dengan metode *canny edge detector* untuk mendapatkan tepi objek makanan. Setelah itu dilakukan penjumlahan piksel yang bernilai 1 sebagai nilai perimeter. Implementasi proses hitung perimeter ditunjukkan pada Kode Program 5.14.

```
1   def perimeter(self, img_bw):
2       h, w = img_bw.shape[:2]
3       bw = np.zeros((h + 2, w + 2), np.uint8)
4       for x in range(h+2):
5           for y in range(w+2):
6               if ((x > 0 and y > 0) and (x < h + 1 and y < w
    + 1)):
7                   bw[x][y] = img_bw[x - 1][y - 1]
8       edges = cv2.Canny(bw, 0, 1)
9       edges[edges > 0] = 1
10      P = 0
11      for x in range(h+2):
12          for y in range(w+2):
13              P += edges[x][y]
14      return P
```

Kode Program 5.14 Proses Hitung Perimeter

Penjelasan dari Kode Program 5.14 Proses Hitung Perimeter adalah sebagai berikut.

Baris 2 : menyimpan panjang dan lebar citra
Baris 3 : inisialisasi array bernilai 0 yang berukuran 2 piksel lebih panjang dan 2 piksel lebih lebar
Baris 4-7 : perulangan untuk memindah nilai citra img_bw ke array bw yang ukurannya lebih lebar
Baris 8 : deteksi tepi dengan canny
Baris 9 : mengubah citra hasil deteksi tepi sehingga bernilai 1 atau 0
Baris 10 : inisialisasi variabel P bernilai 0 untuk menampung nilai perimeter
Baris 11-13 : perulangan untuk menghitung nilai perimeter dengan cara menjumlahkan setiap piksel pada citra

5.2.2.7 Implementasi Proses Hitung *Aspect Ratio*

Implementasi proses hitung *aspect ratio* membutuhkan masukan nilai *major axis length* dan *minor axis length.* Proses hitung aspect ratio dilakukan dengan menghitung nilai perbandingan *major axis length* dan *minor axis length* sesuai

Persamaan 2.14. Implementasi proses hitung aspect ratio dapat dilihat pada Kode Program 5.15.

```
1   def aspect_ratio(self, L, W):
2       AR = float(L) / W
3       return AR
```

Kode Program 5.15 Proses Hitung *Aspect Ratio*

Penjelasan dari Kode Program 5.15 Proses Hitung *Aspect Ratio* adalah sebagai berikut.
Baris 1 : deklarasi method `aspect_ratio`
Baris 2 : menghitung nilai AR dengan cara membagi L dan W
Baris 3 : mengembalikan nilai AR

5.2.2.8 Implementasi Proses Hitung *Roundness*

Implementasi proses hitung *roundness* membutuhkan masukan nilai area dan perimeter. Proses hitung *roundness* dilakukan sesuai Persamaan 2.15. Implementasi proses hitung *roundness* dapat dilihat pada Kode Program 5.16.

```
1   def roundness(self, A, p):
2       R = (4 * math.pi * A) / (p ** 2)
3       return R
```

Kode Program 5.16 Proses Hitung *Roundness*

Penjelasan dari Kode Program 5.16 Proses Hitung *Roundness* adalah sebagai berikut.
Baris 1 : deklarasi method roundness dengan parameter A, L dan W
Baris 2 : menghitung nilai roundness sesuai masukan nilai area dan perimeter
Baris 3 : mengembalikan nilai roundness

5.2.2.9 Implementasi Proses Hitung Rectangularity

Implementasi proses hitung *rectangularity* membutuhkan masukan nilai area, *major axis length* dan *minor axis length.* Proses hitung *rectangularity* dilakukan sesuai Persamaan 2.16. Implementasi proses hitung *roundness* dapat dilihat pada Kode Program 5.17.

```
1   def rectangularity(self, A, L, W):
2       N = float(A) / (L * W)
3       return N
```

Kode Program 5.17 Proses Hitung *Rectangularity*

Penjelasan dari Kode Program 5.17 Proses Hitung *Rectangularity* adalah sebagai berikut.
Baris 1 : deklarasi method rectangulari dengan parameter A, L dan W
Baris 2 : menghitung nilai rectangularity sesuai masukan nilai A, L dan W
Baris 3 : mengembalikan nilai rectangularity

5.2.2.10 Implementasi Proses Hitung *Compactness*

Implementasi proses hitung *compactness* membutuhkan masukan nilai area dan perimeter. Proses hitung *compactness* dilakukan sesuai Persamaan 2.17. Implementasi proses hitung *compactness* dapat dilihat pada Kode Program 5.18.

```
1   def compactness(self, A, P):
2       C = float(P) / math.sqrt(A)
3       return C
```

Kode Program 5.18 Proses Hitung *Compactness*

Penjelasan dari Kode Program 5.18 Proses Hitung *Compactness* adalah sebagai berikut.
Baris 1 : deklarasi method compactness dengan parameter area dan perimeter
Baris 2 : menghitung nilai compactness sesuai masukan nilai area dan perimeter
Baris 3 : mengembalikan nilai compactness

5.2.2.11 Implementasi Proses Hitung *Narrow Factor*

Implementasi proses hitung *narrow factor* membutuhkan masukan nilai diameter dan major axis length. Proses hitung *narrow factor* dilakukan sesuai Persamaan 2.18. Implementasi proses hitung *narrow factor* dapat dilihat pada Kode Program 5.19.

```
1   def narrow_factor(self, D, L):
2       NF = float(D) / L
3       return NF
```

Kode Program 5.19 Proses Hitung *Narrow Factor*

Penjelasan dari Kode Program 5.19 Proses Hitung *Narrow Factor* adalah sebagai berikut.
Baris 1 : deklarasi method narrow_factor dengan parameter D dan L
Baris 2 : menghitung nilai narrow factor (NF) sesuai masukan nilai D dan L
Baris 3 : mengembalikan nilai NF

5.2.2.12 Implementasi Proses Hitung Rasio *Perimeter* dengan *Diameter*

Implementasi proses hitung rasio *perimeter* dengan *diameter* membutuhkan masukan nilai perimeter dan diameter. Proses hitung rasio *perimeter* dengan *diameter* dilakukan sesuai Persamaan 2.19. Implementasi proses hitung rasio *perimeter* dengan *diameter* dapat dilihat pada Kode Program 5.20.

```
1   def ratio_PD(self, P, D):
2       PD = float(P) / D
3       return PD
```

Kode Program 5.20 Proses Hitung Rasio *Perimeter* dan *Diameter*

Penjelasan dari Kode Program 5.20 Proses Hitung Rasio *Perimeter* dan *Diameter* adalah sebagai berikut.
Baris 1 : deklarasi method ratio_PD dengan parameter P dan D

Baris 2 : menghitung nilai rasio perimeter dan diameter (PD) sesuai masukan nilai P dan D
Baris 3 : mengembalikan nilai PD

5.2.2.13 Implementasi Proses Hitung Rasio *Perimeter* dengan *Major Axis Length*

Implementasi proses hitung rasio *perimeter* dengan *major axis length* membutuhkan masukan nilai perimeter dan *major axis length.* Proses hitung rasio *perimeter* dengan *major axis length* dilakukan sesuai Persamaan 2.20. Implementasi proses hitung rasio *perimeter* dengan *major axis length* dapat dilihat pada Kode Program 5.21.

```
1   def ratio_PL(self, P, L):
2       PL = float(P) / L
3       return PL
```

Kode Program 5.21 Proses Hitung Rasio *Perimeter* dengan *Major Axis Length*

Penjelasan dari Kode Program 5.21 Proses Hitung Rasio *Perimeter* dengan *Major Axis Length* adalah sebagai berikut.
Baris 1 : deklarasi method ratio_PL dengan parameter P dan L
Baris 2 : menghitung nilai ratio perimeter dan major axis length (PL) sesuai masukan nilai P dan L
Baris 3 : mengembalikan nilai PL

5.2.2.14 Implementasi Proses Hitung Rasio *Perimeter* dengan *Major* dan *Minor Axis Length*

Implementasi proses hitung rasio *perimeter* dengan *major* dan *minor axis length* membutuhkan masukan nilai perimeter, *major axis length* dan *minor axis length.* Proses hitung rasio *perimeter* dengan *major* dan *minor axis length* dilakukan sesuai Persamaan 2.21. Implementasi proses hitung rasio *perimeter* dengan *major* dan *minor axis length* dapat dilihat pada Kode Program 5.22.

```
1   def ratio_PLW(self, P, L, W):
2       PLW = float(P) / (L + W)
3       return PLW
```

Kode Program 5.22 Proses Hitung Rasio *Perimeter* dengan *Major* dan *Minor Axis Length*

Penjelasan dari Kode Program 5.22 Proses Hitung Rasio *Perimeter* dengan *Major* dan *Minor Axis Length* adalah sebagai berikut.
Baris 1 : deklarasi method ratio_PLW dengan parameter P, L dan W
Baris 2 : menghitung nilai ratio perimeter dan major minor axis length (PLW) sesuai masukan nilai P, L dan W
Baris 3 : mengembalikan nilai PLW

5.2.3 Implementasi Proses Ekstraksi Fitur *Color Moment*

Pada implementasi proses ektraksi fitur *Color Moment* dilakukan pemanggilan seluruh fungsi yang terdapat pada proses tersebut. Fungsi-fungsi

tersebut terdiri dari mean, standar_deviasi, skewness dan kurtosis. Proses tersebut dapat dilihat pada Kode Program 5.23.

```
1   def ColorMoment(self,img_tersegmentasi):
2       mean = self.mean(img_tersegmentasi)
3       std = self.standar_deviasi(img_tersegmentasi,mean)
4       skw = self.skewness(img_tersegmentasi,mean)
5       kurtosis = self.kurtosis(img_tersegmentasi,mean)
6       return (mean[0], mean[1], mean[2], std[0], std[1],
    std[2], skw[0], skw[1], skw[2], kurtosis[0], kurtosis[1],
                kurtosis[2])
```

Kode Program 5.23 Proses Ekstraksi Fitur *Color Moment*

Penjelasan dari Kode Program 5.23 Proses Ekstraksi Fitur *Color Moment* adalah sebagai berikut.

Baris 2-5 : memanggil method/fungsi untuk menghitung masing-masing fitur pada ekstraksi fitur warna

5.2.3.1 Implementasi Proses Hitung *Mean*

Implementasi proses hitung *mean* menggunakan masukan berupa citra yang sudah tersegmentasi. Prosesnya dimulai dengan menjumlahkan semua nilai piksel pada citra tersegmentasi berdasarkan channel R, G dan B. Kemudian hasil penjumlahannya dibagi dengan perkalian panjang dan lebar citra. Implementasi proses hitung *mean* ditunjukkan pada Kode Program 5.23.

```
1   def mean(self,img_tersegmentasi):
2       h, w = img_tersegmentasi.shape[:2]
3       sum = [0, 0, 0]
4       mean = [0, 0, 0]
5       for z in range(3):
6           for x in range(h):
7               for y in range(w):
8                   sum[z] += img_tersegmentasi[x][y][z]
            mean[z] = float(sum[z]) / float(h * w)
9       return (mean)
```

Kode Program 5.24 Proses Hitung *Mean*

Penjelasan dari Kode Program 5.23 Proses Ekstraksi Fitur *Color Moment* adalah sebagai berikut.

Baris 2 : menghitung panjang dan lebar citra tersegmentasi.

Baris 3-4 : inisialisasi variabel sum dan mean untuk menampung nilai penjumlahan dan mean.

Baris 5-8 : perulangan untuk menjumlahkan semua nilai piksel pada citra tersegmentasi.

Baris 9 : membagi nilai penjumlahan dengan panjang dan lebar sebagai nilai mean.

5.2.3.2 Implementasi Proses Hitung *Standard deviation*

Implementasi proses hitung *standard deviation* menggunakan masukan berupa citra yang sudah tersegmentasi dan nilai *mean* untuk setiap channel R, G dan B. Prosesnya dimulai dengan menghitung jumlah selisih setiap piksel dengan *mean*.

berdasarkan channel R, G dan B. Kemudian hasil penjumlahannya dibagi dengan perkalian panjang dan lebar citra. Kemudian hasilnya diakar kuadrat. Implementasi proses hitung *standard deviation* ditunjukkan pada Kode Program 5.25.

```
1   def standar_deviasi(self,img_tersegmentasi,mean):
2       h, w = img_tersegmentasi.shape[:2]
3       diff_std = [0, 0, 0]
4       std = [0, 0, 0]
5       for z in range(3):
6           for x in range(h):
7               for y in range(w):
8                   diff_std[z] +=
    (float(img_tersegmentasi[x][y][z]) - mean[z]) ** 2
9           std[z] = math.sqrt(diff_std[z] / (h * w))
10      return (std)
```

Kode Program 5.25 Proses Hitung *Standard deviation*

Penjelasan dari Kode Program 5.25 Proses Hitung *Standard deviation* adalah sebagai berikut.

Baris 2 : menghitung panjang dan lebar citra tersegmentasi.

Baris 3-4 : inisialisasi variabel diff_std dan std untuk menampung nilai penjumlahan selisih dan *standard deviation*.

Baris 5-8 : perulangan untuk menjumlahkan selisih semua nilai piksel pada citra tersegmentasi dengan mean.

Baris 9 : membagi nilai penjumlahan selisih dengan panjang dan lebar dan melakukan akar kuadrat.

5.2.3.3 Implementasi Proses Hitung *Skewness*

Implementasi proses hitung *skewness* menggunakan masukan berupa citra yang sudah tersegmentasi, nilai *mean* dan nilai *standard deviation*. Proses menghitung nilai *skewness* dilakukan sesuai rumus pada Persamaan 2.24. Implementasi proses hitung *skewness* ditunjukkan pada Kode Program 5.26.

```
1   def skewness(self,img_tersegmentasi,mean,std):
2       h, w = img_tersegmentasi.shape[:2]
3       diff_skw = [0, 0, 0]
4       skw = [0, 0, 0]
5       for z in range(3):
6           for x in range(h):
7               for y in range(w):
8                   diff_skw[z] +=
    (float(img_tersegmentasi[x][y][z]) - mean[z]) ** 3
9           skw[z] = diff_skw[z] / (h * w *(std[z]**3))
10      return (skw)
```

Kode Program 5.26 Proses Hitung *Skewness*

Penjelasan dari Kode Program 5.26 Proses Hitung *Skewness* adalah sebagai berikut.

Baris 2 : menghitung panjang dan lebar citra tersegmentasi.

Baris 3-4 : inisialisasi variabel diff_skw dan skw untuk menampung nilai penjumlahan selisih dan skewness.

Baris 5-8 : perulangan untuk menjumlahkan selisih semua nilai piksel pada citra tersegmentasi dengan mean.

Baris 9 : membagi nilai penjumlahan selisih dengan panjang dan lebar serta *standard deviation* pangkat 3.

5.2.3.4 Implementasi Proses Hitung *Kurtosis*

Implementasi proses hitung kurtosis menggunakan masukan berupa citra yang sudah tersegmentasi, nilai *mean* dan nilai *standard deviation*. Proses perhitungan nilai kurtosis menggunakan rumus pada Persamaan 2.25. Implementasi proses hitung kurtosis ditunjukkan pada Kode Program 5.27.

```
1   def kurtosis(self,img_tersegmentasi,mean,std):
2       h, w = img_tersegmentasi.shape[:2]
3       diff_kurtosis = [0, 0, 0]
4       kurtosis = [0, 0, 0]
5       for z in range(3):
6           for x in range(h):
7               for y in range(w):
8                   diff_kurtosis[z] +=
    (float(img_tersegmentasi[x][y][z]) - mean[z]) ** 4
9           kurtosis[z] = diff_kurtosis[z] / (h * w *
    (std[z]**4))
10      return (kurtosis)
```

Kode Program 5.27 Proses Hitung *Kurtosis*

Penjelasan dari Kode Program 5.27 Proses Hitung *Kurtosis* adalah sebagai berikut.

Baris 2 : menghitung panjang dan lebar citra tersegmentasi.

Baris 3-4 : inisialisasi variabel diff_kurtosis dan kurtosis untuk menampung nilai penjumlahan selisih dan kurtosis.

Baris 5-8 : perulangan untuk menjumlahkan selisih semua nilai piksel pada citra tersegmentasi dengan mean.

Baris 9 : membagi nilai penjumlahan selisih dengan panjang dan lebar serta *standard deviation* pangkat 4.

5.2.4 Implementasi Proses Normalisasi

Implementasi proses normalisasi dimulai dengan melakukan perulangan untuk mendapatkan nilai maksimum dan nilai minimum untuk setiap fitur. Kemudian nilai maksimum dan minimum tersebut akan digunakan untuk proses perhitungan normalisasi sesuai pada Persamaan 2.26. Implementasi proses normalisasi ditunjukkan pada Kode Program 5.28.

```
1   def normalisasi(self,fitur):
2       normalisasi = []
3       max = []
4       min = []
5       for a in range(1, len(fitur[0])):
6           max.append(-1000000)
7           min.append(1000000)
8       for x in range(1, len(fitur[0])):
9           for y in range(0,len(fitur)):
10              if (float(fitur[y][x]) < min[x-1]):
```

```
11                  min[x-1] = float(fitur[y][x])
12              if(float(fitur[y][x])>max[x-1]):
13                  max[x-1] = float(fitur[y][x])
14      for i in range(0,len(fitur)):
15          norm = []
16          norm.append(fitur[i][0])
17          for j in range(1, len(fitur[i])):
18              norm.append((float(fitur[i][j]) - min[j-1]) /
    ((float(max[j-1])) - min[j-1]))
19          normalisasi.append(norm)
20      return normalisasi
```

Kode Program 5.28 Proses Normalisasi

Penjelasan dari Kode Program 5.28 Proses Normalisasi adalah sebagai berikut.

Baris 2 – 4 : deklarasi variabel normalisasi, min max

Baris 5 – 7 : perulangan untuk mengisi list max dengan nilai -1000000 dan min dengan nilai 1000000

Baris 8 – 13 : perulangan untuk menemukan nilai minimal dan maksimal dari isi list fitur

Baris 14 – 19 : perulangan untuk menghitung nilai normalisasi dari setiap nilai fitur.

5.2.5 Implementasi Proses Pengukuran Kemiripan

Implementasi proses pengukuran kemiripan antara data uji dengan setiap *corpus* citra dilakukan dengan menggunakan metode *euclidean distance*. Prosesnya dimulai dengan melakukan perulangan untuk menghitung kuadrat selisih nilai fitur data uji dengan nilai fitur *corpus* citra. Kemudian hasilnya akan diakar kuadrat sebagai nilai *distance*. Proses tersebut dilakukan untuk semua *corpus* citra. Implementasi proses pengukuran kemiripan ditunjukkan pada Kode Program 5.29.

```
1   def euclideanDistance(self, uji):
2       eucli_dist = {}
3       for x in range(len(self.data_training)):
4           sum = 0
5           for y in range(1,len(self.data_training[x])):
6               sum += (float(self.data_training[x][y]) -
    float(uji[y])) ** 2
7   eucli_dist.update({self.data_training[x][0]:math.sqrt(sum)})
8       return eucli_dist
```

Kode Program 5.29 Proses Pengukuran Kemiripan

Penjelasan dari Kode Program 5.29 Proses Pengukuran Kemiripan adalah sebagai berikut.

Baris 2 : deklarasi variabel eucli_dist untuk menampung nilai distance

Baris 3 : perulangan sejumlah data training

Baris 5 – 6 : perulangan sejumlah data fitur untuk menghitung nilai distance

Baris 7 : memasukkan nilai distance ke dalam variabel eucli_dist

BAB 6 PENGUJIAN DAN ANALISIS

Bab ini akan menjelaskan mengenai pengujian dan analisis pada pencarian resep makanan berdasarkan citra makanan menggunakan ekstraksi fitur *Simple Morphological Shape Descriptors* dan *Color Moment*. Terdapat dua pengujian yang dilakukan terhadap penelitian ini antara lain pengujian variasi *k-rank* dan pengujian kombinasi fitur *Simple Morphological Shape Descriptors* dan *Color Moment*.

6.1 Pengujian Variasi Nilai *k-rank*

Parameter yang digunakan pada pengujian ini adalah nilai *k-rank* atau banyaknya resep makanan yang dikembalikan sebagai hasil pencarian. Pengujian ini bertujuan untuk mengetahui pengaruh variasi nilai *k-rank* terhadap nilai *Mean Average Precision*.

6.1.1 Skenario Pengujian Nilai *k-rank*

Pengujian nilai *k-rank* pada pencarian dilakukan menggunakan nilai *k-rank* yang bervariasi. Berikut variasi nilai *k-rank* yang digunakan antara lain 10, 25, 50, 75 dan 100. Contoh hasil pencarian dengan *query* citra Gambar 6.1 untuk nilai *k-rank* 10 ditunjukkan pada Hasil nilai *Mean Average Precision* dapat dilihat pada Tabel 6.2.

Gambar 6.1 Contoh *Query* Pencarian

Tabel 6.1 Contoh Hasil Pencarian dengan *k-rank* = 10

No.	Citra Makanan	Resep Makanan	Keterangan
1.	Hasil Pencarian 1 	**Donat Kentang** **Bahan:** 250 gr kentang, 500 gr tepung terigu, 2 butir telur, 50 gr margarin, 11 gram ragi instant, 100 ml air, 50 gr gula pasir, 1 sdt BPDA, 1/2 sdt garam, 1 sdm susu bubuk **Cara Membuat:** 1. Kupas kentang, rebus, haluskan. Sisihkan. 2. Campur air, gula pasir dan ragi instant. Aduk rata. Diamkan sekitar 15 menit. Bila campuran berbusa maka ragi masih aktif. 3. Campur terigu, BPDA, garam dan subuk. Campur rata. Tambahkan cairan ragi. Aduk rata sampai berbulir. 4. Masukkan kentang yang sudah dihaluskan. Aduk merata. 5. Kocok telur sebentar saja menggunakan garpu. Masukkan ke campuran terigu. Masukkan juga margarin. Uleni sampai kalis. Terus uleni sampai sekitar 40 menit untuk hasil yang maksimal.	Relevan

		6. Taruh adonan di wadah yang agak besar, tutup dengan menggunakan plastic wrap atau serbet bersih. Biarkan untuk proofing sekitar 30-50 menit. 7. Setelah mengembang, kempiskan adonan. Bagi sesuai kebutuhan (kalo saya, saya timbang per adonan sekitar 30-33 gram). Bentuk adonan sesuai selera. Goreng sebentar dalam minyak panas sampai kekuningan. Angkat. Tiriskan. 8. Bila ingin menggunakan gula halus, segera gulingkan donat di gula dalam keadaan panas agar gula menempel. 9. Bila ingin diberi topping glaze atau dcc cair, tunggu sampai donat benar-benar dingin.	
2.	Hasil Pencarian 2 	**Donat Kentang** **Bahan:** 250 gr tepung terigu, 1 butir telur, 1/2 sachet ragi, 1 bungkus susu bubuk, 100 gr kentang kukus, 1 sdm mentega, 50 gr gula pasir, 50 ml air es, 1/2 sdt garam **Cara Membuat:** 1. Masukkan tepung, kentang, gula,telur,susu,ragi aduk kemudian masukkan air es aduk kembali 2. Masukkan mentega dan garam uleni sampai kalis 3. Tutup adonan dengan plastik hingga rapat kemudian Diamkan selama 1 jam sampai 2x lipat mengembang 4. Pukul adonan kemudian bentuk bulat dan bentuk bolong dibagian tengah kemudian diamkan lagi 15 menit 5. Panaskan minyak api jangan terlalu besar,goreng donat hingga kuning kecoklatan sekali balik saja setiap sisi supaya tidak terlalu banyak menyerap minyak 6. Angkat dinginkan dan beri toping sesuai selera	Relevan
3.	Hasil Pencarian 3 	**Donat Kentang Empuk, Lembut, dan Praktis** **Bahan:** 1 kg kentang (direbus lalu dihaluskan), 1 kg tepung terigu, 1/2 kg gula pasir, 6 butir telur, 1 bungkus forvita, 100 gr mentega Blueband, secukupnya Minyak sayur utk menggoreng **Cara Membuat:** 1. Campur semua bahan jadi 1 (kecuali terigu), lalu aduk merata. 2. Masukkan terigu sedikit demi sedikit, uleni hingga adonan kalis (spt digbr). 3. Bentuk bulatan2 donat hingga adonan habis. 4. Siapkan wajan yg sdh diisi minyak, lalu goreng donat hingga menguning, jgn lupa dibolak balik. 5. Donat kentang super lembut, empuk, dan enak sdh jd..tinggal dikasih toping aja sesuai selera	Relevan
4.	Hasil Pencarian 4 	**Donat Kentang Empuk** **Bahan A:** 275 gr tepung terigu, 26 gr susu bubuk, 6 gr ragi instant, 2 butir telur, 35 gr gula pasir, 100 gr kentang kukus, haluskan, 45 ml air **Bahan B :** 43 gr mentega, Sejumput garam Cara Membuat: 1. Campur Bahan A kecuali air masukkan sedikit demi sedikit, sampai teksturnya cukup 2. Uleni sampai kalis 3. Masukkan bahan B 4. Uleni sampai kalis elastis	Relevan

		5. Bulatkan, tutup dengan kain lembab/plastik wrap, diamkan selama 30-40menit 6. Kempiskan adonan, bagi adonan @40gr atau sesuai selera, punya saya jadi 16pc 7. Bulat2kan dan diamkan selama 20menit. 8. Ingat dari yg pertama dibulatkan.. 9. Panaskan minyak 10. Lubangi dengan jari dan goreng sampai matang. 11. Balikkan cukup sekali dan minyak jgn terlalu banyak kalau mau timbul white ringnya.	
5.	Hasil Pencarian 5 	**Ayam Kentucky a.k.a Fried Chicken** **Bahan:** 1/2 kg daging ayam, 1/4 kg tepung terigu protein tinggi, 2 sdm maizena, 1 bks royco, 1 sdt soda kue, 1 gelas air es **Bumbu:** 4 siung bawang putih, 1 ruas kunyit, 1 ruas jahe, 1 sdt ketumbar, secukupnya Garam **Cara Membuat:** 1. Haluskan bumbu, lumurkan ke ayam hingga rata. Simpan dalam kulkas semalaman. 2. Campur tepung-tepungan dan royco. Sisihkan. Dalam wadah terpisah: air es diberi soda kue dan 3 sdm campuran tepung. Simpan dalam kulkas. 3. Keesokan harinya: cemplungkan ayam ke dalam adonan kering, lalu celup ke adonan es lalu cemplungkan lagi ke adonan kering sambil dicubit-cubit lalu celup lagi ke adonan es, cemplung lagi ke tepung dan cubit sampai mendapatkan tingkat kekruilan yang diinginkan. Kalau saya rata-rata 3x celup cemplung sudah cukup ngruil. 4. Goreng dengan minyak banyak dan panas menggunakan api kecil agar matangnya sampai ke dalam. Goreng hingga kecoklatan. 5. Angkat dan tiriskan. Hidangkan hangat.	Tidak relevan
6.	Hasil Pencarian 6 	**Ayam kribo / fried chicken ala kf*** **Bahan:** Bahan 1 (pelapis kering): 450 grm tepung proting (saya pakai cakra kembar), 150 grm tepung maizena, Secukupnya kaldu/garam (dirasa asinnya pas) Bahan 2: 1 ekor ayam ukuran besar (sisihkan kepala dan kaki), 10 siung bawang putih, 1 siung bawang merah, 1 sdt ketumbar bubuk, 1,5 sdt kunyit bubuk, 1 ruas jahe, 3 siung cabe jablai, 1 sdt garam Bahan 3 (pelapis basah): 200 ml air es, 1 butir telur kocok lepas + sejumput garam, 1 sdt soda kue **Cara Membuat:** 1. Cuci bersih ayam. Blender semua bahan 2 (kecuali ayam). Balur semua bumbu ke ayam dan masukkan kedalam kulkas semalaman. 2. Buat adonan pelapis dari adonan 1, campur semua bahan hingga rata, sisihkan 3. Buat adonan pelapis basah dari bahan 3, campur semua dan tambahkan 5 sdm dari bahan 1 (pelapis kering), aduk rata 4. Keluarkan ayam dari kulkas yg sudah semalaman kemudian masukkan kedalam bahan pelapis basah, kemudian masukkan ke bahan pelapis kering sambil	Tidak relevan

		diremas dan dicubit-cubit berulang kali, kalau ingin kribo lagi masukkan ke air es kemudian masukkan lagi ke lapisan kering 5. Siapkan wajan cekung (saya pakai panci ??) dengan minyak yg banyak (deep fry) supaya ayam matang sampai kedalam. Masukkan ayam dan goreng hingga kecoklatan 6. Angkat dan sajikan	
7.	Hasil Pencarian 7 	**Telor Dadar Pedas** **Bahan:** 3 buah telur ayam, Daun bawang iris, 2 siung Bawang merah, 1 siung Bawang putih, 4 buah Cabai, Garam, Kaldu bubuk **Cara Membuat:** 1.Goreng bawang dan cabai, lalu haluskan. 2.Campur telur, daun bawang, bumbu halus, garam dan kaldu bubuk, kocok merata. 3.Goreng tipis sesuai selera.	Tidak relevan
8.	Hasil Pencarian 8 	**Dadar Telor Tahu** **Bahan:** 3-4 butir telor ayam, 2 buah tahu, 1 batang bawang prey, iris, 2 buah tomat sedang, potong dadu kecil, 1/2 sdt garam, 1/4 sdt kaldu jamur, Sedikit minyak **Cara Membuat:** 1.Didalam wadah, Hancurkan/lumatkan tahu dengan sendok. 2.Tambahkan garam dan kaldu jamur, aduk rata. 3.Tambahkan telur ayam, aduk rata. Masukkan bawang prey dan tomat, aduk rata. 4.Masukkan ke penggorengan yang sudah dipanasin terlebih dahulu dan diberi sedikit minyak. Kalau saya biasa menggunakan wajan Happy call, jadi langsung masuk semua adonannya. 5.Masak sampai matang di satu sisi, balik ke sisi yg lainnya masak sampai matang.	Tidak relevan
9.	Hasil Pencarian 9 	**Keripik kentang rumahan** **Bahan:** 3 buah kentang,potong tipis, 250 ml air untuk merendam kentang, secukupnya Garam **Cara Membuat:** 1. Panaskan minyak,goreng kentang,hingga kecoklat2 an 2. Gampang banget kan bunda..... selamat mencoba	Tidak relevan
10.	Hasil Pencarian 10 	**Rendang Daging Kering** **Bahan:** 1/2 daging sapi, 1 butir kelapa untuk santan, 1/2 butir kelapa untuk disangrai, 1 ons cabe merah keriting, 5 siung bawang putih, 7 siung bawang merah, 1 ruas jari jahe, 1 ruas jari lengkuas, 2 batang serai ambil putihnya,-1/2 sdt lada bubuk, 1/2 sdt ketumbar, Sedikit biji pala, Sedikit jinten, secukupnya Daun salam, secukupnya Daun jeruk, secukupnya Daun kunyit, secukupnya Gula merah, secukupnya Garam **Cara Membuat:** 1. Potong daging sapi ukuran sedang 2. Sangrai 1/2 kelapa hingga coklat lalu giling hingga mengeluarkan minyak. 3. Buat santan dari 1 butir kelapa dipisah kental dan cairnya	Tidak relevan

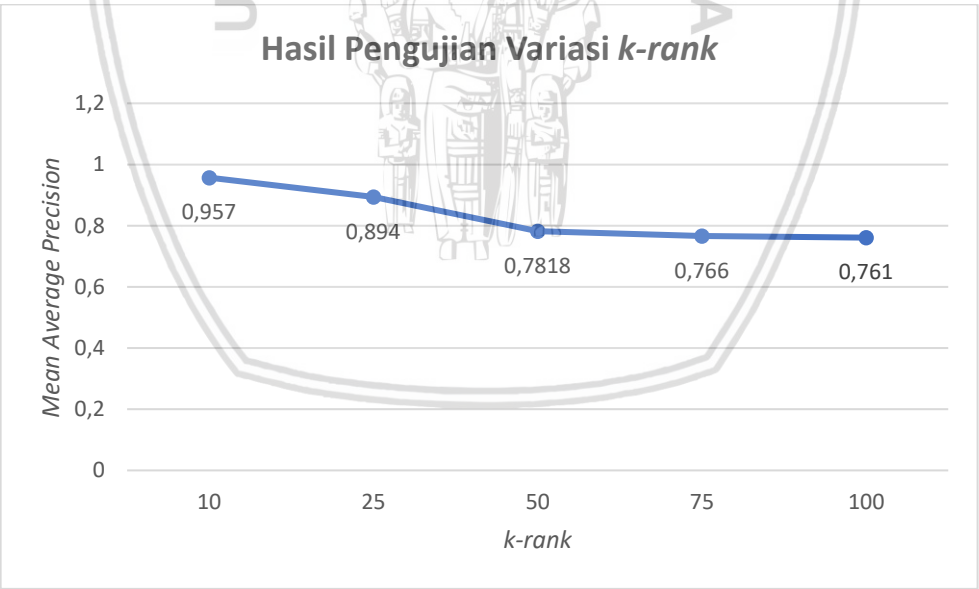
		4. Blender bumbu bawang putih,merah,cabe,jahe,lengkuas,sereh,lada,ketumbar,biji pala,jinten hingga halus 5. Masukkan santan cair dengan bumbu yg sudah diblender panaskan.jika sudah tercium bau wangi,masukkan daun salam,jeruk dan kunyit 6. Jika sudah agak berkurang air santannya masukkan daging sapi lalu aduk 7. Jika sudah terlihat menyusut santannya,masukkan kelapa yg sudah disangrai tadi,lalu masukkan garam dan gula,aduk hingga rata lalu masukkan santan kental dan royconya..aduk terus dengan api kecil...keloreksi rasa,bila sudah pas aduk hingga santan menyusut dan rendang sudah terlihat kering...	

Tabel 6.2 Hasil Nilai MAP pada masing-masing *k-rank*

k-rank	**100**	**75**	**50**	**25**	**10**
MAP	0,76108	0,76661	0,78186	0,89423	**0,95713**

Tabel 6.2 menunjukkan nilai MAP terbaik pada *k-rank* 10 yaitu 95,71%. Nilai MAP akan semakin menurun ketika nilai *k-rank* ditambah. Terlihat bahwa nilai MAP mengalami penurunan drastis pada *k-rank* 50, dengan nilai MAP sebesar 0,78186 dari nilai MAP sebesar 0,89423 ketika pada *k-rank* 25.

6.1.2 Analisis Pengujian Nilai *k-rank*



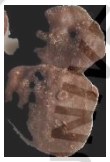
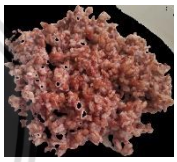
Gambar 6.2 Grafik MAP pada tiap *k-rank*

Berdasarkan Tabel 6.2, hasil pengujian yang telah dilakukan maka akan disajikan dalam bentuk grafik pada Gambar 6.2 untuk menjeaskan pengujian yang telah dilakukan. Pada gambar tersebut terlihat nilai MAP tergantung pada nilai *k-rank* yang mana MAP akan semakin baik jika nilai *k-rank* yang digunakan semakin kecil. Semakin besar nilai *k-rank* yang digunakan menyebabkan semakin banyak hasil pencarian yang tidak relevan karena pada setiap kelas hanya memiliki *corpus*

berjumlah 8. Nilai MAP terbaik diperoleh saat nilai *k-rank* dikecilkan menjadi *k-rank* 10 dengan MAP sebesar 95,713%. Dapat disimpulkan bahwa pada *k-rank* 10 sistem mengembalikan hasil yang baik sesuai kebutuhan pengguna.

Pada *k-rank* 100, 75 dan 50 menghasilkan nilai MAP kecil karena ada beberapa *query* yang memiliki nilai *Average Precision* dibawah 0,5. Hal tersebut dikarenakan hasil *pre-processing* kurang sempurna. Hasil *pre-processing* untuk beberapa citra masih menyertakan *background* yang seharusnya terhapus. Hal ini berpengaruh pada hasil ekstraksi fitur bentuk *Simple Morphological Shape Descriptors* karena hasil ekstraksi menghasilkan bentuk yang berbeda dan hasil ekstraksi fitur warna *Color Moment* karena nilai piksel-piksel citra berbeda. Contoh citra tersebut ditunjukkan pada Tabel 6.3.

Tabel 6.3 Contoh Citra dengan Hasil *Pre-processing* Tidak Bagus

Citra Asli				
Citra hasil *pre-processing*				

Proses *resize* pada *pre-processing* juga turut memengaruhi hasil ekstraksi fitur. Yang mana untuk citra yang berukuran asimetris ketika dilakukan *resize* sehingga berukuran simetris (500x500 piksel) menghasilkan bentuk objek makanan yang berbeda. Perbedaan *resize* citra antara citra yang semula simetris dengan asimetris ditunjukkan pada Gambar 6.3.

(a) Citra asimetris

(b) Hasil *resize* citra asimetris

(c) Hasil *pre-processing* citra asimetris

(d) Citra simetris

(e) Hasil *resize* citra simetris

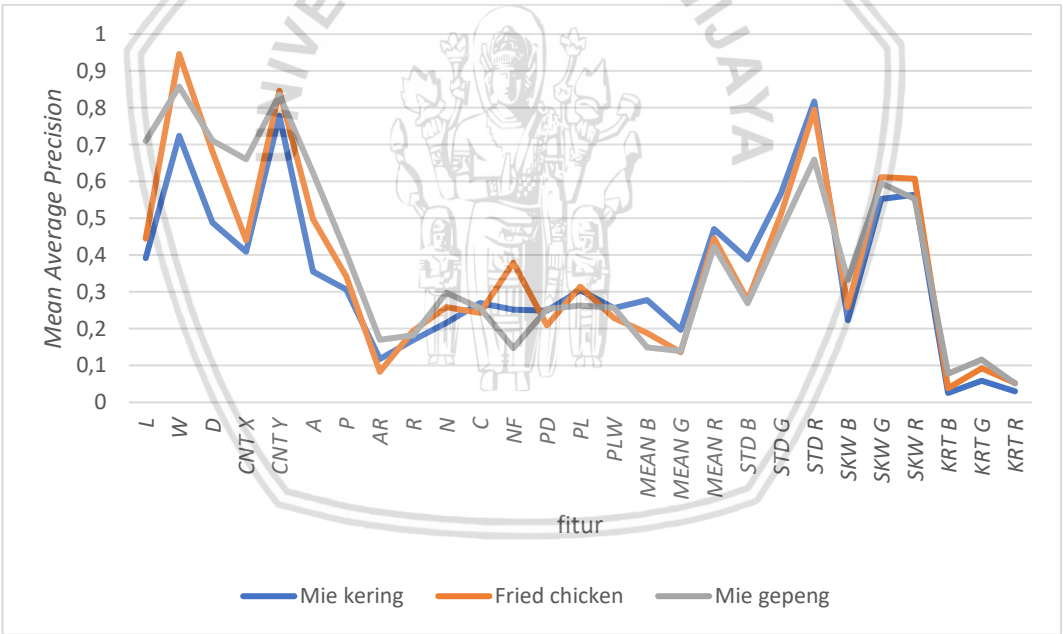
(f) Hasil *pre-processing* citra simetris

Gambar 6.3 Contoh Hasil *Resize* Citra

Faktor lainnya yaitu beberapa jenis makanan memiliki warna yang sama. Misalnya mie kering, fried chicken dan mie kriting yang memiliki warna serupa yaitu coklat. Gambar 6.4 menunjukkan jenis makanan yang memiliki warna serupa. Ketika dilakukan ekstraksi fitur, nilai fitur *Color Moment* menghasilkan nilai yang hampir sama. Hal tersebut ditunjukkan pada Gambar 6.5 yang mana grafik untuk fitur Color Moment memiliki garis yang berhimpitan.

(a) mie kering (b) fried chicken (c) mie kriting

Gambar 6.4 Jenis Makanan Berbeda yang Memiliki Warna Serupa



Gambar 6.5 Grafik Nilai Fitur Jenis Makanan Berbeda dengan Warna Serupa

6.2 Pengujian Kombinasi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*

Pengujian ini bertujuan untuk mengetahui pengaruh penggunaan fitur *Simple Morphological Shape Descriptors* dan *Color Moment* terhadap hasil pencarian resep makanan.

6.2.1 Skenario Pengujian Kombinasi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*

Pengujian ini dilakukan dengan menghitung nilai MAP pada penggunaan fitur *Simple Morphological Shape Descriptors* saja, fitur *Color Moment* saja dan kombinasi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment* pada setiap *k-rank* 10. Hasil pengujian ini ditunjukkan pada Tabel 6.4.

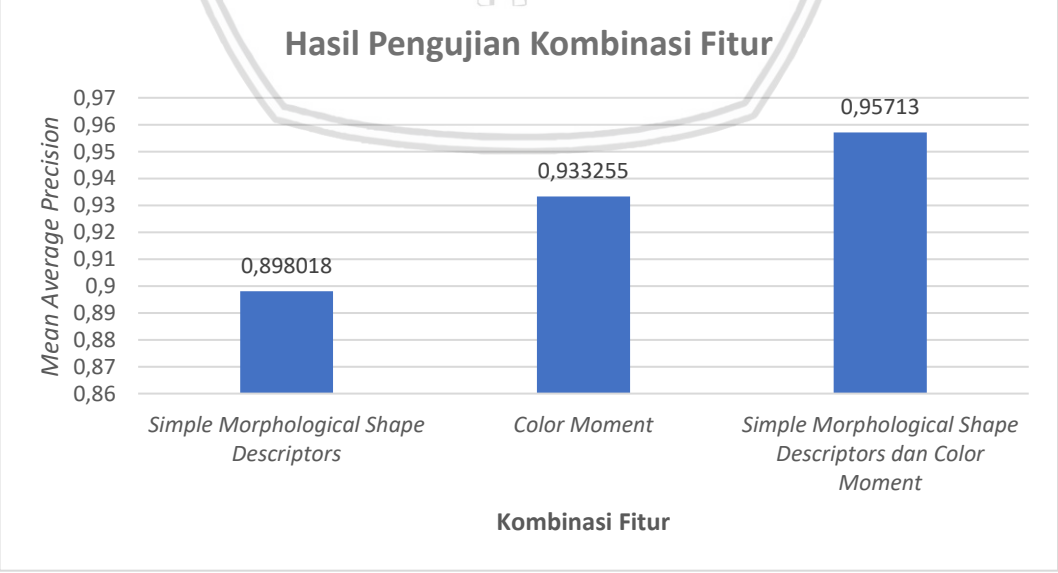
Tabel 6.4 Hasil Pengujian Kombinasi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*

Kombinasi Fitur	*Simple Morphological Shape Descriptors*	*Color Moment*	*Simple Morphological Shape Descriptors* dan *Color Moment*
MAP	0,898018	0,933255	**0,95713**

Tabel 6.4 menunjukkan penggunaan kombinasi fitur *Simple Morphological Shape Descriptors* dan *Color Moment* berpengaruh terhadap hasil MAP. Nilai MAP terendah didapatkan pada penggunaan fitur *Simple Morphological Shape Descriptors* saja yaitu 89,8%. Sedangkan nilai MAP terbesar didapatkan pada penggunaan kombinasi fitur *Simple Morphological Shape Descriptors* dan *Color Moment* yaitu sebesar 95,71%.

6.2.2 Analisis Pengujian Kombinasi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment*

Berdasarkan Tabel 6.4, hasil pengujian yang telah dilakukan maka akan disajikan dalam bentuk grafik pada Gambar 6.6 untuk menjelaskan pengujian yang telah dilakukan.



Gambar 6.6 Grafik MAP pada Pengujian Kombinasi Fitur

Grafik pada Gambar 6.6 menunjukkan penggunaan fitur *Color Moment* saja menghasilkan nilai MAP lebih besar dari pada penggunaan fitur *Simple Morphological Shape Descriptors* saja. Penggunaan fitur *Color Moment* terbukti lebih baik dari pada penggunaan fitur *Simple Morphological Shape Descriptors*. Gabungan fitur *Simple Morphological Shape Descriptors* dan *Color Moment* mampu meningkatkan nilai MAP hasil pencarian. Hal ini menunjukkan penggunaan salah satu fitur *Simple Morphological Shape Descriptors* dan *Color Moment* saja tidak mampu untuk membedakan beberapa jenis makanan yang memiliki bentuk atau warna hampir sama. Dengan menggabungkan kedua fitur dapat mengatasi kelemahan dari penggunaan masing-masing fitur.

Penggunaan fitur *Simple Morphological Shape Descriptors* kurang baik dikarenakan perbedaan sudut pandang saat memotret citra makanan seringkali menghasilkan nilai fitur bentuk yang berbeda. Misalnya untuk citra makanan pada Gambar 6.7 yang terdiri dari dua citra fried chicken yang berbeda sudut pandang pengambilan citra dan citra mie gepeng. Kemudian Masing-masing fitur bentuk dari ketiga citra makanan tersebut dibandingkan yang ditunjukkan pada Tabel 6.5. Hasilnya antara citra fried chicken 1 dengan citra lainnya, dari 15 fitur bentuk *Simple Morphological Shape Descriptors* 8 fitur menunjukkan citra fried chicken 1 lebih dekat ke citra mie gepeng. Delapan fitur tersebut yaitu *major axis length*, *minor axis length*, *centroid* X, *centroid* Y, area, *aspect ratio*, *narrow factor,* dan rasio perimeter dengan diameter. Selain itu perbandingan antara citra Fried chicken 2 dengan citra lainnya, 8 fitur bentuk juga menunjukkan citra fried chicken 2 lebih dekat ke Mie gepeng. Fitur- fitur tersebut diantaranya *major axis length*, *minor axis length*, *centroid* X, *centroid* Y, area, *aspect ratio*, *rectangularity*, dan *narrow factor*. Setelah dikalkulasi untuk keseluruhan fitur, citra Fried chicken 1 dan Fried chicken 2 lebih mirip citra Mie gepeng dari pada sesama citra Fried chicken.

(a) Fried chicken 1 (c1) (b) Fried chicken 2 (c2) (c) Mie gepeng (c3)

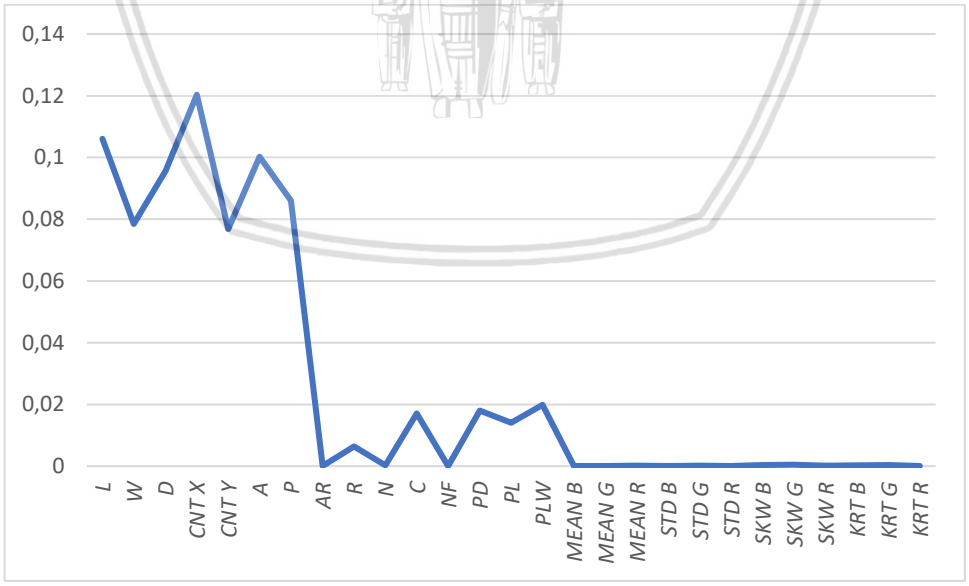
Gambar 6.7 Contoh Citra Makanan dengan Sudut Pandang Pengambilan Berbeda

Tabel 6.5 Jarak Citra Makanan dengan Sudut Pandang Pengambilan Berbeda

Fitur	Fried Chicken 1 (c1)	Fried Chicken 2 (c2)	Mie Gepeng (c3)	Jarak Fried Chicken 1		Jarak Fried Chicken 2	
				d(c1,c2)	d(c1,c3)	d(c2,c1)	d(c2,c3)
L	0.454	0.918	0.659	0.2150	0.0420	0.2150	0.0669
W	0.954	0.470	0.644	0.2346	0.0964	0.2346	0.0302
D	0.689	0.702	0.562	0.0002	0.0162	0.0002	0.0195

Cnt X	0.447	0.868	0.755	0.1776	0.0950	0.1776	0.0128
Cnt Y	0.858	0.443	0.625	0.1720	0.0543	0.1720	0.0331
A	0.505	0.483	0.492	0.0005	0.0001	0.0005	0.0001
P	0.350	0.269	0.486	0.0065	0.0185	0.0065	0.0469
AR	0.084	0.403	0.224	0.1019	0.0198	0.1019	0.0318
R	0.189	0.271	0.102	0.0067	0.0076	0.0067	0.0285
N	0.251	0.323	0.355	0.0052	0.0108	0.0052	0.0010
C	0.248	0.181	0.379	0.0045	0.0172	0.0045	0.0394
NF	0.376	0.002	0.070	0.1399	0.0933	0.1399	0.0047
PD	0.213	0.140	0.391	0.0053	0.0316	0.0053	0.0628
PL	0.318	0.105	0.349	0.0452	0.0009	0.0452	0.0592
PLW	0.234	0.168	0.401	0.0043	0.0280	0.0043	0.0541
Euclidean Distance				1.0579	0.7292	1.0579	0.7008

Gambar 6.8 Contoh Citra Makanan Dipotret dengan Ketinggian Berbeda



Gambar 6.9 Grafik Varians Citra Makanan Dipotret dengan Ketinggian Berbeda

Selain itu ketinggian pemotretan objek makanan juga turut memengaruhi nilai fitur *Simple Morphological Shape Descriptors* sedangkan fitur Color Moment tidak terpengaruh. Gambar 6.8 menunjukkan citra makanan yang memiliki kelas

sama yaitu mie gepeng yang dipotret dengan ketinggian berbeda. Pada gambar tersebut terlihat objek makanan yang dipotret dengan jarak lebih dekat memiliki dimensi lebih besar. Ukuran dimensi citra yang berbeda menyebabkan nilai fitur *Simple Morphological Shape Descritors* yang berbeda sedangkan fitur *Color Moment* tidak terpengaruh karena warnanya tetap sama. Hal tersebut juga dibuktikan dengan nilai varians setiap fitur yang ditunjukkan pada Gambar 6.9. Nilai varians fitur *Simple Morphological Shape Descriptors* cenderung lebih besar dibandingkan fitur *Color Moment* yang semua nilainya mendekati 0. Nilai varians yang semakin besar menunjukkan data tersebut semakin beragam padahal kedua gambar tersebut memiliki kelas yang sama.

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan pengujian dan analisis yang telah dilakukan, didapatkan kesimpulan sebagai berikut.

1. Berdasarkan pengujian variasi nilai *k-rank* didapat nilai MAP tertinggi pada *k-rank* 10 yaitu 95,713% dan nilai MAP terendah pada k-rank 100 yaitu 76,108%. Semakin besar nilai *k-rank* berpengaruh negatif pada nilai MAP yang dihasilkan. Hal ini disebabkan karena semakin banyak hasil pencarian yang dikembalikan maka akan semakin banyak hasil yang tidak relevan karena *corpus* yang digunakan pada setiap kelas sedikit.
2. Pengujian penggunaan kombinasi fitur menunjukkan penggunaan fitur *Color Moment* saja menghasilkan nilai MAP 93,32% sedangkan fitur *Simple Morphological Shape Descriptors* saja menghasilkan nilai MAP 89,8%. Penggunaan fitur *Color Moment* terbukti lebih baik dari pada penggunaan fitur *Simple Morphological Shape Descriptors*. Hal ini dikarenakan perbedaan sudut pandang dan perbedaan ketinggian pengambilan citra sangat memengaruhi hasil ekstraksi fitur *Simple Morphological Shape Descriptors*. Penggunaan gabungan fitur *Simple Morphological Shape Descriptors* dan *Color Moment* mampu meningkatkan nilai MAP hasil pencarian menjadi 95,71%. Hal ini menunjukkan penggunaan salah satu fitur *Simple Morphological Shape Descriptors* dan *Color Moment* saja tidak mampu untuk membedakan beberapa jenis makanan yang memiliki bentuk atau warna hampir sama. Dengan menggabungkan kedua fitur dapat mengatasi kelemahan dari penggunaan masing-masing fitur.

7.2 Saran

Adapun saran yang diberikan untuk penelitian selanjutnya adalah sebagai berikut.

1. Perlu dilakukan seleksi fitur untuk menyeleksi fitur-fitur yang kurang relevan sehingga dapat meningkatkan akurasi pencarian.
2. Perlu dilakukan penelitian lebih lanjut terkait metode ekstraksi fitur yang digunakan misalnya menggunakan ekstraksi fitur tekstur mengingat fitur ini tidak dipengaruhi oleh sudut pandang pengambilan objek makanan.
3. Perlu dilakukan penelitian lebih lanjut terkait citra makanan yang digunakan misalnya untuk makanan yang sudah dimakan sebagian serta untuk makanan yang terdiri dari berbagai jenis.

DAFTAR PUSTAKA

Aakif, A. & Faisal, M., 2015. Automatic Classification of Plants Based on Their Leaves. *Biosystems Engineering,* Volume 139, pp. 66-75.

Afifa, Z., 2016. Implementasi Metode Gaussian Filter untuk Penghapusan Noise Pada Citra Menggunakan GPU.S1. *Universitas Islam Negeri Maulana Malik Ibrahim.*

Agaputra, M. D., Wardani, K. R. R. & Siswanto, E., 2013. Pencarian Citra Digital Berbasiskan Konten dengan Ekstraksi Fitur HSV, ACD, dan GLCM. *Jurnal Telematika,* 8(2), pp. 8-13.

Ambarwati, A., passarella, r. & Sutarno, S., 2017. Segmentasi Citra Digital Menggunakan Thresholding Otsu untuk Analisa Perbandingan Deteksi Tepi. *Annual Research Seminar (ARS),* 2(1), pp. 216-226.

Amynarto, N., Sari, Y. A. & Wihandika, R. C., 2018. Pengenalan Emosi Berdasarkan Ekspresi Mikro Menggunakan Metode Local Binary Pattern. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 2(10), pp. 3230-3238.

Azis, F., 2013. Sistem Temu Kembali Citra Kain Berbasis Tekstur dan Warna.S1. *Universitas Islam Negeri Sultan Kasim Riau.*

Baskoro, S. Y., Ridok, A. & Furqon, M. T., 2015. Pencarian Pasal Pada Kitab Undang-Undang Hukum Pidana (KUHP) Berdasarkan Kasus Menggunakan Metode Cosine Similarity dan Latent Semantic Indexing (LSI). *Journal of Environmental Engineering & Sustainable Technology,* 2(2), pp. 83-88.

Caglayan, A., Guclu, O. & Can, A. B., 2016. *A Plant Recognition Approach Using Shape and Color Features in Leaf Images.* Berlin, Heidelberg, International Conference on Image Analysis and Processing, pp. 1-10.

Chamidah, N., Wiharto & Salamah, U., 2012. Pengaruh Normalisasi Data pada Jaringan Syaraf Tiruan Backpropagasi Gradient Descent Adaptive (BPGDAG) untuk Klasifikasi. *Jurnal ITSMART,* 1(1), pp. 28-33.

Frediansah, Sasongko, P. S. & Endah, S. N., 2012. Sistem Temu Kembali Citra Berbasis Histogram Warna Fuzzy untuk Pencarian Citra Berwarna. *Journal of Informatics and Technology,* 1(1), pp. 130-136.

Irawati & Sugiarti, 2016. *Aplikasi Resep Masakan Berbasis Mobile dengan Menggunakan Metode Forward Chaining.* Makasar, Seminar Nasional Riset Ilmu Komputer, pp. 251-256.

Kaipravan, Muhsina; Rejiram, R, 2016. *A Novel CBIR System Based on Combination of Color Moment and Gabor Filter.* Ernakulam, India, International Conference on Data Mining and Advanced Computing (SAPIENCE), pp. 170-174.

Kusuma, S. F., Pawening, R. E. & Dijaya, R., 2017. Otomatisasi Klasifikasi Kematangan Buah Mengkudu Berdasarkan Warna dan Tekstur. *Jurnal Ilmiah Teknologi Sistem Informasi,* 3(1), pp. 17-23.

Lestari, S. Y. & Kursini, 2015. Membangun Aplikasi Mobile Resep Masakan Asia (Indonesia,China,Jepang) Berbasis Android. *Jurnal Data Manajemen dan Teknologi Informasi (DASI),* 13(1), pp. 36-41.

Mazid, K., 2013. Segmentasi Citra Daun Tembakau Berbasis Deteksi Tepi Menggunakan Algoritma Canny.S1. *Universitas Dian Nuswantoro.*

Nugraheny, D., 2015. Metode Nilai Jarak Guna Kesamaan Atau Kemiripan Ciri Suatu Citra (Kasus Deteksi Awan Cumulonimbus Menggunkan Principal Component Analysis). *Jurnal Angkasa,* 2(2), pp. 21-30.

Prasad, S., Peddoju, S. K. & Ghosh, D., 2013. *Mobile Plant Species Classification : A Low Computational Aproach.* Shimla, India, IEEE Second International Conference on Image Information Processing (ICIIP-2013), pp. 405-409.

Primahayu, R. A., Utaminingrum, F. & Syauqy, D., 2017. Sistem Monitoring Cairan Infus Terpusat Menggunakan Pengolahan Citra Digital. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 1(8), pp. 649-657.

Rejito, J., 2013. Pengklasteran K-Means Database Citra Untuk Meningkatkan Akurasi Pencarian Query CBIR Menggunakan Intensitas Warna. *Jurnal Matematika Integratif,* 9(1), pp. 91-107.

Sandy, W. K., Widodo, A. W. & Sari, Y. A., 2018. Penentuan Klasifikasi Tanda Tangan Menggunakan Shape Features Extraction Techniques Dengan Metode Klasifikasi K Nearest Neighbor dan Mean Average Precision. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* 2(3), pp. 1083-1091.

Sari, Y. A., Dewi, R. K. & Fatichah, C., 2014. Seleksi Fitur Menggunakan Ekstraksi Fitur Bentuk, Warna, dan Tekstur dalam Sistem Temu Kembali Citra Daun. *JUTI: Jurnal Ilmiah Teknologi Informasi,* 12(1), pp. 1-8.

Sarker, I. H. & Iqbal, S., 2013. Content-based Image Retrieval Using Haar Wavelet Transform and Color Moment. *Smart Computing Review,* 3(3), pp. 155-165.

Setiawan, A., Suryani, E. & Wiharto, 2016. Segmentasi Citra Sel Darah Merah Berdasarkan Morfologi Sel Untuk Mendeteksi Anemia Defisiensi Besi. *Jurnal Teknologi & Informasi ITSmart,* 3(1), pp. 1-8.

Singh, H. & Agrawal, D., 2017. Result Analysis and Comparison of Hybrid Method based on Local Binary Pattern (LBP) and Color Moment (CM) for Efficient Image Retrieval. *International Journal of Computer Applications,* 159(5), pp. 14-19.

Sutarno, Abdullah, R. F. & Passarella, R., 2017. *Identifikasi Tanaman Buah Berdasarkan Fitur Bentuk, Warna dan Tekstur Daun Berbasis Pengolahan Citra dan Learning Vector Quantization (LVQ).* Palembang, Annual Research Seminar, pp. 65-70.

Syafi'i, S. I., Wahyuningrum, R. T. & Muntasa, A., 2015. Segmentasi Obyek Pada Citra Digital Menggunakan Metode Otsu Thresholding. *Jurnal Informatika,* 13(1), pp. 1-8.

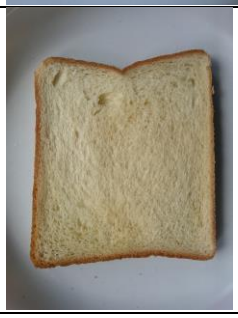
Wäldchen, J. & Mäder, P., 2018. Plant Species Identification Using Computer Vision Techniques : A Systematic Literature Review. *Archives of Computational Methods in Engineering,* Volume 25, pp. 507-543.

Lampiran A *Corpus* Resep Makanan

No.	Nama Makanan	Kode	Citra Makanan	Resep Makanan
1.	Donat	001_1		Donat kentang Bahan: 400 gr tepung terigu, 200 gr kentang yg sudah direbus dan dilunatkan, 1/2 sendok gula putih, Sejumput garam, 4 gram fermipan, 50 ml air, 1 butir telur Cara Membuat: 1. Bikin biangnya dulu ya, masukkan fermipan dan sejumput garam serta air hangat kuku kedalam gelas dan tutup rapat. Tunggu selama 3 menit hingga berbusa. 2. Masukkan telur. Tepung terigu. Garam. Gula. Aduk hingga merata dan uleni hingga kalis. Saya ngulennya pakai tangan aj, tutup adonan selama 1 jam 4. Jika sudah mengembang. Kempiskan adonan dan bulatkan kemudian lubangi tengahnya. Saya ngelubanginya pakai tutup botol, profing kurleb setengah jam 6. Goreng dengan api sedang. Cukup dibalik sekali aja supaya tdk meresap minyak
2.	Donat	001_8		Donat Kentang Empuk Bahan: 275 gr tepung terigu protein tinggi mis. Cakra kembar, 26 gr susu bubuk, 6 gr ragi instant, 2 butir telur, 35 gr gula pasir, 100 gr kentang kukus, haluskan, 45 ml air, 43 gr mentega, Sejumput garam Cara Membuat: 1. Campur Bahan A kecuali air masukkan sedikit demi sedikit, sampai teksturnya cukup 2. Uleni sampai kalis, masukkan bahan B , Uleni sampai kalis elastis 5. Bulatkan, tutup dengan kain lembab/plastik wrap, diamkan selama 30-40menit 6. Kempiskan adonan, bagi adonan @40gr atau sesuai selera, punya saya jadi 16pc 7. Bulat2kan dan diamkan selama 20menit,Ingat dari yg pertama dibulatkan, Panaskan minyak 10. Lubangi dengan jari dan goreng sampai matang. 11. Balikkan cukup sekali dan minyak jgn terlalu banyak kalau mau timbul white ringnya.

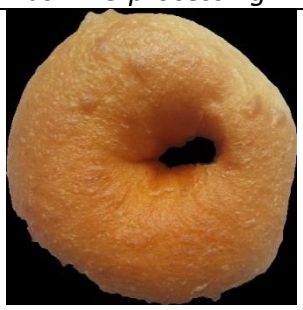
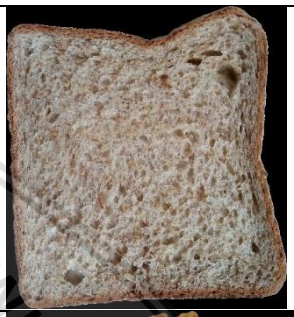
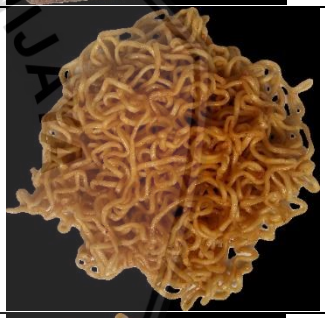
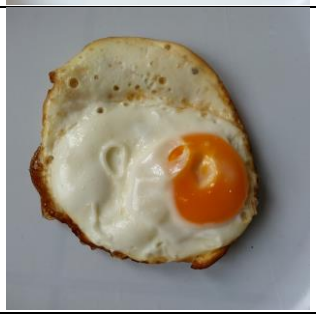
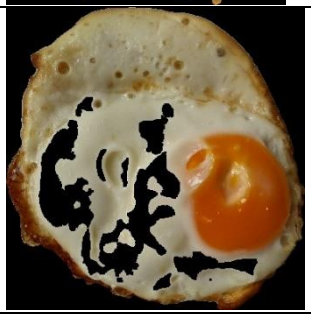
Data selengkapnya dapat dilihat di https://goo.gl/GWsEQB

Lampiran B *Query* Citra Makanan

No.	Nama *Query*	Citra *Query*
1.	Query 1	
2.	Query 2	
3.	Query 3	
4.	Query 4	
5.	Query 5	

Data selengkapnya dapat dilihat di https://goo.gl/qWUZmJ

Lampiran C Hasil *Pre-Processing Corpus* Dan *Query*

No.	Nama	Citra Awal	Hasil *Pre-processing*
1.	Donat		
2.	Roti Gandum		
3.	Mie Kriting		
4.	Mie gepeng		
5.	Telor ceplok		

Dapat dilihat di https://goo.gl/JJiUS9

Lampiran D Hasil Ekstraksi Fitur *Corpus* Dan *Query*

Nama	L	W	D	Cnt_x	Cnt_y	A	P	AR	R	N	C	NF	PD	PL	PLW	X1	X2	X3	δ1	δ1	δ3	S1	S2	S3	K1	K2	K3
corpus\001_1	423.0	416.0	451.8	213.0	209.0	133444.0	1641.0	1.0	0.6	0.8	4.5	1.1	3.6	3.9	2.0	46.9	87.7	132.9	37.4	57.8	79.2	0.5	-0.4	-0.9	2.6	2.0	2.1
corpus\001_10	424.0	420.0	465.2	213.0	209.0	144585.0	1577.0	1.0	0.7	0.8	4.1	1.1	3.4	3.7	1.9	45.9	88.4	135.4	38.7	55.2	72.3	0.6	-0.3	-1.0	2.6	2.1	2.6
corpus\001_2	431.0	421.0	451.4	219.0	210.0	134321.0	1690.0	1.0	0.6	0.7	4.6	1.0	3.7	3.9	2.0	39.1	83.7	130.0	37.7	59.4	81.7	0.8	-0.2	-0.8	2.9	1.9	1.9
corpus\001_3	419.0	407.0	436.8	213.0	202.0	126175.0	1590.0	1.0	0.6	0.7	4.5	1.0	3.6	3.8	1.9	34.9	80.0	128.9	32.9	55.8	80.7	0.8	-0.2	-0.8	2.8	1.9	2.0
corpus\001_5	305.0	298.0	321.2	155.0	147.0	66904.0	1195.0	1.0	0.6	0.7	4.6	1.1	3.7	3.9	2.0	17.5	57.2	112.9	20.4	43.0	72.6	1.5	0.1	-0.7	5.4	2.1	1.9
corpus\001_7	268.0	264.0	282.2	135.0	131.0	52454.0	1070.0	1.0	0.6	0.7	4.7	1.1	3.8	4.0	2.0	14.0	55.2	110.9	18.4	41.2	70.3	1.9	0.1	-0.7	7.7	2.2	1.9
corpus\001_8	270.0	265.0	282.7	136.0	132.0	52779.0	1066.0	1.0	0.6	0.7	4.6	1.0	3.8	3.9	2.0	13.7	52.9	107.1	17.4	39.5	68.5	1.9	0.1	-0.7	8.4	2.2	1.9
corpus\001_9	267.0	263.0	280.0	135.0	130.0	51753.0	1057.0	1.0	0.6	0.7	4.6	1.0	3.8	4.0	2.0	15.2	53.6	109.3	18.6	40.4	70.1	1.7	0.1	-0.7	7.2	2.2	1.9
corpus\002_1	472.0	437.0	583.4	245.0	210.0	173880.0	1678.0	1.1	0.8	0.8	4.0	1.2	2.9	3.6	1.8	101.8	113.3	125.9	57.7	59.6	61.4	-0.6	-0.8	-1.1	2.2	2.5	3.1
corpus\002_10	387.0	453.0	501.8	197.0	203.0	127310.0	1456.0	0.9	0.8	0.7	4.1	1.3	2.9	3.8	1.7	71.4	81.5	95.8	53.4	57.7	63.0	-0.2	-0.4	-0.6	1.6	1.6	1.8
corpus\002_3	473.0	438.0	579.9	244.0	210.0	172931.0	1669.0	1.1	0.8	0.8	4.0	1.2	2.9	3.5	1.8	93.1	102.7	115.8	55.7	57.0	59.3	-0.4	-0.6	-1.0	2.1	2.3	2.8
corpus\002_4	419.0	440.0	531.2	216.0	200.0	143909.0	1521.0	1.0	0.8	0.8	4.0	1.3	2.9	3.6	1.8	81.9	94.4	105.7	55.7	59.4	61.9	-0.3	-0.5	-0.8	1.8	1.9	2.2
corpus\002_6	412.0	409.0	512.0	211.0	183.0	132968.0	1460.0	1.0	0.8	0.8	4.0	1.2	2.9	3.5	1.8	89.6	100.1	114.3	58.8	61.8	65.4	-0.3	-0.5	-0.8	1.8	2.0	2.3
corpus\002_7	415.0	450.0	521.8	212.0	202.0	138994.0	1509.0	0.9	0.8	0.7	4.0	1.3	2.9	3.6	1.7	78.2	89.5	103.5	57.0	61.3	66.0	-0.2	-0.4	-0.6	1.7	1.7	1.9
corpus\002_8	416.0	450.0	520.1	212.0	202.0	139210.0	1507.0	0.9	0.8	0.7	4.0	1.3	2.9	3.6	1.7	77.7	89.2	103.0	57.0	61.3	65.8	-0.2	-0.4	-0.6	1.6	1.7	1.9

Data selengkapnya dapat dilihat di https://goo.gl/CKkZxd

Lampiran E Hasil Normalisasi Fitur *Corpus* Dan *Query*

Nama	L	W	D	Cnt_x	Cnt_y	A	P	AR	R	N	C	NF	PD	PL	PLW	X1	X2	X3	δ1	δ1	δ3	S1	S2	S3	K1	K2	K3
corpus\001_1	0.76	0.80	0.64	0.77	0.80	0.74	0.17	0.20	0.70	0.56	0.04	0.06	0.07	0.07	0.05	0.43	0.52	0.86	0.51	0.71	0.88	0.19	0.40	0.37	0.01	0.12	0.17
corpus\001_10	0.76	0.81	0.67	0.77	0.80	0.81	0.16	0.20	0.82	0.70	0.02	0.08	0.05	0.05	0.03	0.43	0.53	0.88	0.54	0.65	0.74	0.21	0.42	0.34	0.01	0.15	0.25
corpus\001_2	0.78	0.82	0.64	0.81	0.81	0.74	0.18	0.20	0.66	0.52	0.05	0.04	0.08	0.07	0.05	0.36	0.48	0.83	0.52	0.75	0.93	0.23	0.46	0.41	0.02	0.10	0.14
corpus\001_3	0.74	0.78	0.60	0.77	0.76	0.69	0.16	0.20	0.70	0.52	0.04	0.04	0.07	0.06	0.04	0.31	0.44	0.82	0.42	0.67	0.91	0.22	0.44	0.40	0.02	0.10	0.14
corpus\001_5	0.38	0.47	0.28	0.40	0.45	0.31	0.08	0.20	0.66	0.51	0.05	0.05	0.08	0.07	0.05	0.13	0.20	0.67	0.16	0.37	0.74	0.30	0.53	0.43	0.05	0.15	0.13
corpus\001_7	0.27	0.37	0.17	0.28	0.36	0.22	0.06	0.20	0.64	0.52	0.05	0.05	0.08	0.07	0.05	0.10	0.17	0.65	0.12	0.33	0.70	0.34	0.54	0.42	0.08	0.16	0.14
corpus\001_8	0.27	0.37	0.17	0.28	0.36	0.22	0.06	0.20	0.65	0.51	0.05	0.04	0.08	0.07	0.05	0.09	0.15	0.62	0.10	0.29	0.66	0.34	0.53	0.43	0.09	0.16	0.13
corpus\001_9	0.26	0.37	0.17	0.28	0.35	0.21	0.06	0.20	0.65	0.51	0.05	0.04	0.08	0.07	0.05	0.11	0.16	0.64	0.12	0.31	0.69	0.32	0.54	0.43	0.07	0.16	0.13
corpus\002_1	0.91	0.86	1.00	0.97	0.81	1.00	0.18	0.22	0.87	0.78	0.01	0.20	0.01	0.04	0.03	1.00	0.79	0.79	0.94	0.75	0.52	0.08	0.28	0.30	0.01	0.22	0.33
corpus\002_10	0.64	0.91	0.78	0.67	0.77	0.70	0.14	0.14	0.85	0.48	0.02	0.25	0.01	0.06	0.01	0.69	0.45	0.51	0.85	0.71	0.55	0.12	0.41	0.44	0.00	0.05	0.11
corpus\002_3	0.91	0.87	0.99	0.96	0.81	0.99	0.18	0.22	0.88	0.76	0.01	0.19	0.01	0.04	0.03	0.91	0.68	0.70	0.90	0.69	0.47	0.10	0.32	0.35	0.01	0.18	0.29
corpus\002_4	0.74	0.87	0.86	0.79	0.75	0.80	0.15	0.18	0.88	0.62	0.01	0.23	0.01	0.05	0.02	0.80	0.59	0.60	0.90	0.75	0.53	0.11	0.36	0.40	0.00	0.11	0.18
corpus\002_6	0.72	0.78	0.80	0.75	0.65	0.73	0.14	0.20	0.88	0.64	0.01	0.21	0.01	0.04	0.02	0.87	0.65	0.68	0.96	0.80	0.60	0.10	0.35	0.38	0.00	0.12	0.20
corpus\002_7	0.73	0.90	0.83	0.76	0.76	0.77	0.15	0.16	0.86	0.53	0.02	0.22	0.01	0.05	0.01	0.76	0.54	0.58	0.92	0.79	0.61	0.12	0.40	0.44	0.00	0.07	0.13
corpus\002_8	0.74	0.90	0.83	0.76	0.76	0.77	0.15	0.17	0.87	0.53	0.02	0.21	0.01	0.05	0.01	0.75	0.54	0.58	0.92	0.79	0.61	0.12	0.40	0.44	0.00	0.07	0.13
corpus\002_9	0.65	0.92	0.79	0.67	0.78	0.71	0.14	0.14	0.86	0.49	0.02	0.26	0.01	0.06	0.01	0.65	0.41	0.46	0.82	0.67	0.48	0.13	0.43	0.46	0.00	0.05	0.11
corpus\003_1	0.61	0.85	0.89	0.62	0.86	0.81	0.15	0.14	0.88	0.90	0.01	0.38	0.01	0.08	0.04	0.94	0.93	0.91	0.65	0.57	0.33	0.03	0.09	0.10	0.02	0.54	0.75
corpus\003_10	0.45	0.69	0.70	0.42	0.65	0.59	0.11	0.14	0.88	0.92	0.01	0.39	0.01	0.08	0.04	0.93	0.96	0.97	0.62	0.55	0.33	0.03	0.07	0.07	0.02	0.60	0.85
corpus\003_2	0.59	0.82	0.86	0.60	0.82	0.77	0.14	0.15	0.88	0.90	0.01	0.38	0.01	0.08	0.04	0.94	0.93	0.92	0.65	0.57	0.35	0.03	0.09	0.10	0.02	0.54	0.75
corpus\003_4	0.33	0.56	0.54	0.36	0.56	0.43	0.08	0.15	0.89	0.92	0.01	0.38	0.01	0.08	0.04	0.89	0.92	0.94	0.59	0.52	0.31	0.03	0.08	0.07	0.02	0.59	0.85

Data selengkapnya dapat dilihat di https://goo.gl/C5BbvQ

Lampiran F Hasil Pencarian Resep Makanan

Ekstraksi Fitur *Simple Morphological Shape Descriptors* dan *Color Moment* pad *k-rank* = 10

Dapat dilihat di https://goo.gl/UAXL5e